VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

SHLUKOVÁNÍ SLOV PODLE VÝZNAMU

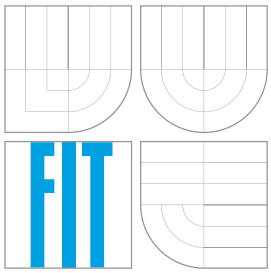
DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. ZBYNĚK JADRNÍČEK

BRNO 2015

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

SHLUKOVÁNÍ SLOV PODLE VÝZNAMU

WORD SENSE CLUSTERING

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. ZBYNĚK JADRNÍČEK

VEDOUCÍ PRÁCE
SUPERVISOR

Doc. RNDr. PAVEL SMRŽ, Ph.D.

BRNO 2015

Abstrakt

Tato práce se zabývá problémem sémantické podobnosti slov v angličtině. Čtenář je nejprve informován o teorii shlukování slov podle významu, poté jsou popsány některé metody a nástroje související s tématem. V praktické části navrhneme a implementujeme systém pro výpočet sémantické podobnosti slov využívající nástroj Word2Vec, konkrétně se zaměříme na biomedicínské texty z databáze MEDLINE. Na závěr práce budeme diskutovat dosažené výsledky a předložíme několik návrhů, jak systém vylepšit.

Abstract

This thesis is focused on the problem of semantic similarity of words in English language. At first reader is informed about theory of word sense clustering, then there are described chosen methods and tools related to the topic. In the practical part we design and implement system for determining semantic similarity using Word2Vec tool, particularly we focus on biomedical texts of MEDLINE database. At the end of the thesis we discuss reached results and give some ideas to improve the system.

Klíčová slova

biomedicínská data, sémantická podobnost, tokenizace, vektorový prostorový model, Word2Vec, zpracování přirozeného jazyka

Keywords

biomedical data, semantic similarity, tokenization, vector space model, Word2Vec, natural language processing

Citace

Zbyněk Jadrníček: Shlukování slov podle významu, diplomová práce, Brno, FIT VUT v Brně, 2015

Shlukování slov podle významu

Prohlášení

Prohlašuji, že jsem tuto práci vypracoval samostatně pod vedením doc. RNDr. Pavla Smrže, Ph.D. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Zbyněk Jadrníček
26. května 2015

Poděkování

Rád bych poděkoval doc. RNDr. Pavlu Smržovi, Ph.D. za vedení, rady a připomínky při tvorbě této práce. Dále děkuji všem svým blízkým za vyjádřenou podporu.

Obsah

Kapitola 1

Úvod

V posledních letech došlo k velkému rozvoji komunikačních technologií, což způsobilo rapidní nárůst objemu informací, které denně vznikají ve formě elektronických dokumentů. Vyhledávání informací v nich, klasifikace a jiné zpracování, se pro člověka stalo časově příliš náročné, a proto roste potřeba vytváření systémů automatického zpracování textu. Ty se však potýkají s řadou problémů vyplývajících z vlastností jazyka.

Jedním z nich je mnohoznačnost (víceznačnost, polysémie). To znamená, že některá slova nebo i delší útvary mohou mít různé významy podle kontextu, ve kterém byla použita. Každý jazyk má různý podíl mnohoznačných slov. Pro češtinu je tento podíl poměrně malý a většinou nedochází k nedorozumění při jejich použití. Naopak k jazykům s častým výskytem vícevýznamových slov patří čínština nebo angličtina. Uvádí se, že v angličtině má 121 nejčastějších podstatných jmen průměrně 7,8 odlišných významů, 70 nejfrekventovanějších sloves dokonce 12, což společně tvoří asi 20 % slov v běžném textu [19].

Člověk se naučí určovat správný význam slova během svého života, avšak pro počítače představuje variabilita jazyka a nejednoznačnost problém. Výrazně komplikuje jejich interakci s člověkem prostřednictvím přirozeného jazyka nebo zhoršuje výsledky internetových vyhledávačů. Vyřešení mnohoznačnosti patří k úkolům v oblasti zpracování přirozeného jazyka (natural language processing, NLP). NLP se zabývá analýzou přirozeného jazyka a jeho pochopením z pohledu počítačů.

Tato práce se věnuje rozpoznávání významově podobných slov jako jednomu z problémů NLP. Metody zjišťující podobnost slov pracují na základě množiny ručně psaných pravidel nebo využívají statistické analýzy výskytu slov v rozsáhlé sbírce textů, tzv. korpusu. Dle domluvy s vedoucím práce, doc. RNDr. Pavlem Smržem, Ph.D., jsem se hlouběji zaměřil na oblast biomedicínských textů. Využil jsem databázi MEDLINE, obsahující miliardy slov v abstraktech publikací.

Nedávný pokrok v oblasti algoritmů učení bez učitele umožnil využití těchto velkých neanotovaných korpusů pro vytváření statistických jazykových modelů. Výsledkem jsou kvalitní reprezentace vektorového prostoru, které mohou být užitečným nástrojem v mnoha úlohách. Cílem této práce je vytvoření modelu, jenž by usnadnil práci s medicínskými daty, jelikož jejich množství v reálném životě roste podobně jako u obecných textů.

Na začátku práce (2) zmíníme některé základní pojmy související s tématem práce, jako je význam slova, NLP, podobnost slov a způsoby jejího výpočtu. Třetí kapitola (3) se od teorie přesunuje k zpracování dokumentů počítačem. Věnuje se krokům předzpracování textu, modelům reprezentace dokumentů a distribučním sémantickým modelům. Na tyto pojmy naváže i následující kapitola (4), protože popisuje nástroj Word2Vec, využívající vektorový model k reprezentaci slov.

Poté (5) se podíváme na porovnávání významů vět, které souvisí s úkolem konference SemEval, jenž jsem se pokusil vyřešit. Jak již bylo zmíněno, cílem práce je vytvořit systém, umožňující vyhledávání synonym v biomedicínských datech. Šestá kapitola (6) se proto zabývá implementací takového systému. Uvedeme také porovnání několika dostupných tokenizátorů, protože biomedicínská doména vyžaduje specifické zpracování (7).

Předposlední kapitola (8) se věnuje vyhodnocení funkčnosti vytvořeného systému, budeme jej porovnávat s dalšími prostředky. Na závěr (9) budou shrnuty dosažené výsledky, zdůrazníme přínos práce a navrhneme možnosti další činnosti.

Kapitola 2

Podobnost slov

Pro pochopení této práce je nezbytné, aby byly vysvětleny některé teoretické pojmy, které budou používány v dalších kapitolách. Nejprve se zaměříme na popis slova a jeho významu, blíže se podíváme na obor NLP. Poté definujeme sémantickou podobnost dvou slov a ukážeme některé matematické metody, jak ji určit. Nakonec vysvětlíme pojem shlukování slov podle významu. Tato kapitola má za úkol uvést čtenáře do problematiky, její pochopení nevyžaduje hlubší znalost informačních technologií.

2.1 Slovo a jeho význam

Slovo je skupina hlásek s určitým významem, tvořící ustálený celek, jednotku slovní zásoby. Význam označuje to, co slovo nebo delší text míní, co vyjadřuje. Slovo může mít dva významy:

- slovní (obsahový, lexikální) – označuje význam, který má slovo samo o sobě, vyjadřuje jazykově ztvárněný odraz skutečnosti ve vědomí mluvčího (*lev* – silné zvíře se čtyřma nohama, hřívou, velkou tlamou atd.)
- mluvnický (gramatický) – slovo jej získá, když vyjadřuje různé gramatické kategorie (rod, číslo, čas atd.), tedy když je ve spojení s jinými slovy ve větě (*lev*, *lvem* – mají různý mluvnický význam, ale neliší se významem slovním)

V užití těchto významů se jednotlivé jazyky liší. Zatímco některé se v dané situaci vyjadřují pomocí gramatických prostředků, jiné použijí prostředky lexikální. Podle významu slova rozlišujeme tyto 3 typy slov:

- lexikální – mají jen slovní význam (např. příslovce)
- lexikálně gramatická (plnovýznamová) – mají význam slovní a ve větě také různé významy mluvnické (např. podstatná jména)
- gramatická (neplnovýznamová) – plný význam získávají až ve spojení s jiným slovem, nejsou větnými členy, vyjadřují větné vztahy slov nebo vět (např. spojky)

Některá slova přirozeného jazyka mají více než jednu interpretaci, v takovém případě mluvíme o slově mnohoznačném. Samotný význam u takových slov lze rozdělit podle způsobu vzniku na:

- základní (primární, původní) – *noha* jako končetina živočichů
- druhotný (sekundární, odvozený) – *noha* jako součást nábytku, pojmenování na základě podobnosti

S mnohoznačností souvisí i pojmy polysémie a homonymie.

- polysémie – mnohoznačnost, kde významy slova mají jistou podobnost (již zmíněné slovo *noha*)
- homonymie – mezi významy neexistuje žádná logická spojitost (*rys* – zvíře, znak).

Polysémie se netýká pouze slov, ale i jiných textových útvarů, zejména pokud jde o výpovědi vytržené z kontextu. Větnou polysémii způsobují:

- homonymní nebo polysémní výrazy (*Ztratil korunu.*)
- víceznačné syntaktické konstrukce (*Vlastní starý dům a automobil.* – Automobil je také starý?)

Vznik homonym bývá nahodilý, nejčastěji k němu dochází odvozením slov od podobných základů (*vinný* – vytvořen z révy, někdo, komu byla prokázána vina) nebo přijímáním z cizích jazyků (*kolej* – rýha např. po projetí kola blátem, místo ubytování studentů). Hranice mezi homonymií a polysémií nemusí být vždy jednoznačná. V širším pojetí se za homonyma považují i slova, jejichž významy sice původně vznikly na základě mnohovýznamovosti, ale v současnosti již není vzájemná souvislost zřejmá.

Plnovýznamová a lexikální slova jsou z hlediska této práce nejdůležitější, protože jejich lexikální význam (dále často jen význam) budeme v dalších kapitolách porovnávat. Pro tuto sekci jsem použil informace z knihy [23].

2.2 Zpracování přirozeného jazyka

Zpracování přirozeného jazyka (dále NLP) je obor výpočetního modelování nejrůznějších aspektů jazyka a tvorby širokého spektra systémů, určených pro strojový překlad, rozpoznávání řeči, porozumění jazyku či jeho tvorbě. Takové systémy mohou poskytovat rozhraní webovým vyhledávačům, databázím nebo také expertním systémům. Zpracovávání textu a systémy porozumění obsahu jsou užitečné při extrakci informací z textů a jeho formátování mnoha způsoby pro další aplikace.

Můžeme říct, že NLP propojuje lingvistiku s informatikou. Vývoj konkrétního systému často vyžaduje znalost z dalších domén, jako jsou společenské vědy, statistika, psychologie nebo medicína, jedná se tedy o mezidisciplinární obor. NLP hraje významnou roli při naší komunikaci se stroji. Význam určitého slova můžeme popsat jedním z následujících způsobů:

- přirozeným jazykem – slovo je popsáno nějakým vhodným způsobem pomocí slov již známých, tento způsob používáme při běžné komunikaci, pokud chceme vysvětlit lexikální význam ostatním lidem
- formálním jazykem – tento typ dokáže počítač lépe zpracovat, používá vhodné teoretické nástroje, jako jsou gramatiky nebo kalkuly, naopak pro komunikaci mezi lidmi je takový způsob nevhodný

Počítače tedy vyžadují komunikaci v jazyce, který je přesný, jednoznačný a vysoce strukturovaný, podobně jako programovací jazyk. Přirozený jazyk je z tohoto pohledu nepoužitelný. Jak již bylo uvedeno, lidská řeč není vždy přesná, obsahuje mnohoznačnost, závisí na slangu a sociálním prostředí. To je důvod, proč vývoj řady NLP aplikací není jednoduchý. Jinou možností je zadání pokynu prostřednictvím omezené množiny jasně formulovaných hlasových pokynů.

NLP systémy se obecně věnují syntaktické a sémantické analýze. Na syntaktické úrovni jsou vytvářeny algoritmy k rozpoznávání mluvnických kategorií v dané větě, typicky se jedná o určování slovních druhů a dalších vztahů. První takové systémy byly vytvořeny pro angličtinu, ale nepracovaly spolehlivě nebo nebyly schopny reagovat na vývoj jazyka. V současnosti je proto vyvíjeno mnoho vícejazyčných algoritmů, které jsou vůči těmto změnám odolné. Do sémantické části spadají např. algoritmy získávání znalostí, k extrakci frází nebo shlukování termínů odkazujících ke stejné entitě [22].

Dřívější implementace zahrnovaly ručně psané kódy množiny pravidel, avšak moderní NLP systémy jsou založeny na strojovém učení, kde bylo použito mnoho odlišných tříd algoritmů. V zásadě mají na vstupu velkou množinu dat, která slouží k natrénování modelu obsahujícího automaticky vytvořená pravidla. V dalším kroku je tento model otestován a je vyhodnocena jeho úspěšnost. Testování a trénování musí probíhat na odlišných datech, jinak dojde ke zkreslení výsledku.

Některé z prvních použitých algoritmů, jako byly rozhodovací stromy, produkovaly systémy pravidel `if-then` podobných těm s ručně psanými pravidly [29]. Později se výzkum zaměřil na statistické modely, které prováděly pravděpodobnostní rozhodování založené na přiřazování vah každé vstupní jednotce. Procedura učení použitá během strojového učení se automaticky zaměřuje na nejčastější případy, zatímco pro ruční psaní pravidel často není zřejmé, čemu věnovat pozornost. Přesnost systémů založených na automatickém učení pravidel může být vylepšena jednoduše poskytnutím dalších vstupních dat.

2.3 Sémantická podobnost

Sémantická podobnost slov je metrika, která dané dvojici slov přiřadí číselnou hodnotu v závislosti na jejich lexikální podobnosti. Vyšší hodnota značí větší podobnost. Skóre sémantické podobnosti produkované běžnými metodami nabývá normalizované hodnoty z intervalu $\langle 0, 1 \rangle$, kde 1 značí úplnou podobnost, tedy totožnost. Např. slova *kůň* a *kráva* považujeme do jisté míry za sémanticky podobná, protože obě popisují hospodářská zvířata. Metody měření podobnosti většinou vychází z těchto tří pravidel [12]:

- Podobnost mezi objekty A a B závisí na tom, co mají společného. Čím více toho spolu sdílejí, tím jsou podobnější.
- Podobnost mezi objekty A a B je inverzně závislá na odlišnostech mezi nimi. Čím více mají rozdílů, tím méně podobné jsou.
- Maximální podobnosti mezi objekty A a B by mělo být dosaženo pouze pokud jsou identické.

Sémantickou podobnost lze počítat i pro fráze, věty nebo celé dokumenty. Bývá označována také jako sémantická blízkost nebo vzdálenost, blízká slova mohou být např. i antonyma. Jedním z důvodů, proč je určování podobnosti významu složité, je, že neexistuje exaktní hodnota, která by mohla být považovaná jako jediná za správnou. Kontrola výsledků často vyžaduje jistou míru lidské znalosti z dalších oborů.

2.3.1 Metody výpočtu podobnosti

Existují různé matematické nástroje k odhadu sémantických souvislostí mezi jednotkami jazyka. Podobnost slov můžeme určovat z uspořádání topologie, např. v orientovaném acyklickém grafu bychom použili délku nejkratší cesty mezi dvěma uzly. V této práci se zaměříme na reprezentaci slov pomocí vektorů, proto popíšu některé metriky jejich podobnosti. Nejdůležitější z nich je kosinová podobnost, protože ji používá řada nástrojů.

Manhattanská vzdálenost

$$distance_{manh}(\vec{x}, \vec{y}) = \sum_{i=1}^{n} |x_i - y_i| \tag{2.1}$$

Euklidovská vzdálenost

$$distance_{eucl}(\vec{x}, \vec{y}) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \tag{2.2}$$

Kosinová podobnost

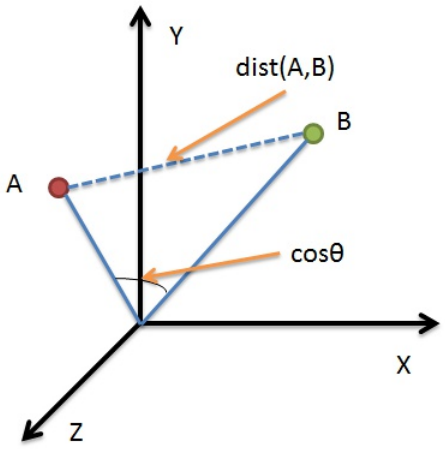
$$sim_{cos}(\vec{x}, \vec{y}) = \frac{\vec{x}\,\vec{y}}{|\vec{x}||\vec{y}|} = \frac{\sum_{i=1}^{n} x_i \cdot y_i}{\sqrt{\sum_{i=1}^{n} x_i^2 \sum_{i=1}^{n} y_i^2}} \tag{2.3}$$

Jaccardova podobnost

$$sim_{jacc}(\vec{x}, \vec{y}) = \frac{\sum_{i=1}^{n} min(x_i, y_i)}{\sum_{i=1}^{n} max(x_i, y_i)} \tag{2.4}$$

Diceův koeficient

$$sim_{dice}(\vec{x}, \vec{y}) = \frac{2 \cdot \sum_{i=1}^{n} min(x_i, y_i)}{\sum_{i=1}^{n} (x_i + y_i)} \tag{2.5}$$



Obrázek 2.1: Kosinová vzdálenost dvou bodů v prostoru [20].

2.4 Shlukování slov podle významu

Shlukování slov podle významu je proces identifikace významů, které jsou natolik podobné, že je můžeme považovat za stejné, vzájemně související, případně za významy silně podřazené nějakému obecnějšímu významu. Tento proces může být ruční nebo automatický. Výsledkem shlukování jsou clustery, množiny obsahující sloučená slova reprezentované jedním společným významem.

Tohoto zjednodušení využívají aplikace, které těží z nižšího stupně polysémie, jako je strojový překlad, kde je lexikální mnohoznačnost často uchována mezi některými dvojicemi a příliš citlivé rozlišování významu je zbytečné. Metody shlukování jsou hojně používané při řešení mnoha úkolů ve zpracování přirozeného jazyka s cílem zachytit neznámé kategorie slov. Může být aplikováno i pro zjednoznačňování významu slova. Hlavní myšlenkou tohoto přístupu je, že shlukování objevuje slova sémanticky příbuzná, a tudíž je snadnější určit správný význam v daném kontextu.

Kapitola 3

Reprezentace a zpracování dokumentů

Nyní propojíme teorii z předchozí kapitoly s informatikou. Jak jsme si řekli, na vstupu algoritmů strojového učení bývá obvykle velký objem dat uložených v korpusu. Než tento korpus použijeme k trénování modelů, je nezbytné jej vhodně upravit tak, aby výsledky nezhoršovaly různé typy chyb. Velká část kapitoly se proto bude věnovat předzpracování dat. Poté se podíváme na modely, které používáme k reprezentaci textových dokumentů. Z vektorové reprezentace vychází distribuční sémantické modely, odhadující význam slova dle kontextů jeho výskytu. Od této kapitoly budeme uvádět příklady slov z angličtiny.

3.1 Korpus

Korpus je velké množství uložených textů, které primárně slouží k jazykovému výzkumu. Může obsahovat např. články z novin, populární literaturu, vědecké práce, lékařské záznamy, komunikaci ze sociálních sítí, nebo v případě mluveného jazyka přepis záznamu rozhovoru. Shromáždit sbírku textů z internetu je pochopitelně snadnější, ovšem tento text bývá více zašuměný než ostatní zdroje. Na druhou stranu web pokrývá široké spektrum témat. Dle jazyků, v kterých jsou napsány jednotlivé texty, můžeme korpusy dělit na:

- monolingvní – všechny texty v korpusu jsou v jednom jazyce
- multilingvní – korpus obsahuje texty v různých jazycích

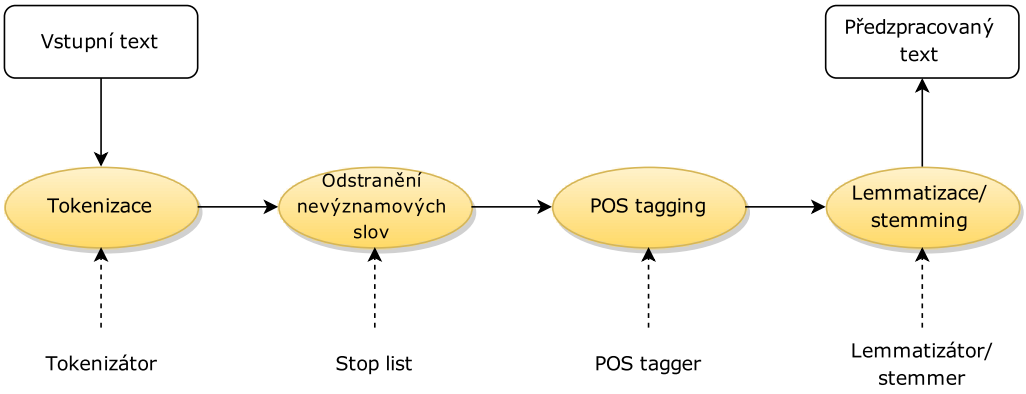
Primární funkcí korpusu je umožnit analýzu širokého množství textů nástrojům v NLP, avšak může sloužit i jako pomůcka pro uživatele, kteří se chtějí naučit novému jazyku. Práci s korpusem často usnadňuje speciální program, jenž umožňuje snadné vyhledávání slov a frází, zjištění jejich frekvence v korpusu nebo nalezení původního textového zdroje. U některých korpusů lze třídit nálezy podle abecedy a také určit jejich slovní druh.

Z tohoto důvodu bývají korpusy anotovány, což dále usnadňuje činnost některých algoritmů strojového učení. Jedná se o proces doplňování gramatických či jiných kategorií do korpusu, které jsou jsou důležité z hlediska řešeného problému. Samotné anotování představuje jednu z komplikací v NLP, protože anotace musí často doplnit člověk. Korpusy mohou být anotovány na mnoha úrovních. Většinou potřebujeme informaci o syntaxi, sémantice a další údaje, tyto nároky však omezují celkovou velikost korpusu [27].

3.2 Předzpracování textu

Nyní se zaměříme na proces zpracování korpusu dříve, než jej použijeme ve strojovém učení. Nejprve musíme získat slovní jednotky z dokumentů a aplikovat na ně několik normalizačních metod. Cílem předzpracování je modifikace vstupního textu takovým způsobem, aby došlo k jeho zkvalitnění a usnadnil tak práci v dalších krocích. Různé jazyky vyžadují odlišné techniky předzpracování, protože se liší gramatickou a morfologickou stavbou.

Předzpracování můžeme rozdělit na tokenizaci, odstraňování nevýznamových slov, part of speech tagging, lematizaci a stemming, viz obr. 3.1. Některé systémy nemusí využívat všechny kroky, jiné používají další techniky (např. vážení). Také není pevně dáno pořadí, v řadě případů může být vhodné odstraňovat nevýznamová slova až po určení slovních druhů.



Obrázek 3.1: Etapy předzpracování vstupního textu.

Kvalita předzpracování má zásadní vliv na přesnost porovnávání. Jedná se ovšem o složitý problém a to zejména z následujících důvodů:

- dat je velké množství
- data jsou z různých zdrojů, nekonzistence dat
- data mohou mít nízkou kvalitu
- je třeba učinit určitá rozhodnutí
- odlišné nástroje vyžadují různé formáty dat

3.2.1 Tokenizace

Nejprve je text rozdělen na seznam elementárních jednotek, tzv. tokenů. Pro věty v psané podobě se jedná o na první pohled jednoduchý problém – slova jsou oddělena mezerou. Komplikace způsobuje spojovník mezi slovy, pro která je třeba identifikovat, jestli je budeme chápat jako jednoslovný výraz nebo je rozdělíme na více tokenů. Podobně musíme rozlišit zkratky od slov s tečkou, která ukončuje větu.

Samostatným problémem v NLP je identifikace víceslovných výrazů, kde je třeba vyřešit několik otázek. Zda je důležitě pořadí slov v sousloví nebo jestli musí jednotlivá slova následovat bezprostředně po sobě. Chybným postupem může být za termín označen i víceslovný výraz, který ve skutečnosti termín není. Někdy se proto za účelem zlepšení výsledků

používá slovník sousloví. U čísel máme několik možných variant zpracování. Můžeme je úplně ignorovat, pracovat s nimi jako se samostatnými slovy, nebo je zpracovat jako prvky závislé na jiném slově.

3.2.2 Odstranění nevýznamových slov

V tomto kroku jsou odstraněna nevýznamová slova, tedy ta, která jsou součástí věty, ale nemají žádný lexikální význam. Jedná se např. o spojky, částice, v angličtině často se vyskytující členy, nebo interpunkci, pokud nebyla odstraněna již při tokenizaci. Tato slova se označují jako tzv. stop slova a bývají uvedena ve stop listu [10].

Jinou možností je využití informace z vážení (bude popsáno níže). Nejčastější termíny s malou nebo žádnou sémantickou hodnotou vzhledem k doméně dokumentů můžeme odstranit a zařadit je do stop listu, avšak vždy musíme brát v úvahu celý kontext. Odstraňování nevýznamových slov je nicméně nepovinné a některé systémy jej úplně vynechávají.

3.2.3 Part of speech tagging

Značení slovních druhů (Part of Speech Tagging, POS Tagging, POST) je další z úkolů zpracování přirozeného jazyka, představující důležitý mezikrok i v jiných úlohách. Jeho cílem je identifikace druhů, jako podstatná jména, slovesa a další, pro každé slovo ve větě. Můžeme jej kombinovat s odstraňováním nevýznamových slov, kde využijeme informaci o jednotlivých slovních druzích. Existují dva přístupy k vytváření programů řešících tento problém, tzv. taggerů [5]:

- založený na pravidlech – k určení značek používají ručně psaná pravidla (např. Brill's tagger)
- stochastický – mnohoznačnost řeší používáním trénovacího korpusu k výpočtu pravděpodobnosti, že k danému slovu patří určitá značka v daném kontextu, často používají skryté Markovovy modely

Na vstupu algoritmu je daná věta a konečná množina značek, které mohou být přiřazeny. Výstupem je nejlépe odpovídající značka pro každé slovo věty na základě jeho definice i kontextu slova. Nástroje pro určování značek můžeme rozdělit na dva typy. První přiřazuje slovům jejich větné členy (podmět, předmět atd.), druhý typ určuje funkcionální role (podstatné jméno, sloveso atd.).

3.2.4 Lemmatizace a stemming

Lemmatizace vrací základní tvar slova, tj. lemma. Tato operace se používá také ve vyhledávačích a dovoluje vyhledávat bez ohledu na konkrétní tvar. Aby vyhledávání fungovalo správně, musí se zpracovat nejen slova v dokumentech, ale stejným způsobem i slova v dotazu. Program, který provádí lemmatizaci, nazýváme lemmatizátor. Podobnou operaci, stemming, vykonává stemmer.

Stemming je na rozdíl od lemmatizace operace, která pro vyskloňované nebo časované slovo (případně odvozené) vrátí kmen slova. Existuje několik přístupů k implementaci stemmeru. Nejjednodušší metodou je odstraňování známých předpon a přípon slov, ovšem pak může nastat situace, kdy nesouvisející slova budou zkrácená na stejný základ. Pokud nastává změna při ohýbání v kořeni, tak tento jednoduchý algoritmus neurčí kořen správně. Dalším

problémem je ukončení kmene morfologickou koncovkou, kde algoritmus může nesprávně utrhnout tuto koncovku a vrátit nesprávný kratší kmen.

Jiné metody využívají slovníku kmenů nebo určují základní tvary statisticky na základě různosti po sobě následujících písmen ve slovech, kde se pomocí frekvence jednotlivých shluků písmen stanoví, zda se jedná o prefix, kmen, nebo sufix.

3.3 Modely reprezentace dokumentů

V této sekci budou popsány tři modely používané k reprezentaci dokumentů, jelikož na jejich základě jsou definovány způsoby určování podobnosti slov. Zmíníme klady a zápory těchto reprezentací, zvláštní pozornost bude věnována vektorovému modelu, protože toho se týká celá práce.

3.3.1 Booleovský model

Jedná se o nejjednodušší z uvedených modelů, založený na teorii množin a Booleově algebře. Dokumenty a dotazy jsou reprezentovány jako množiny termínů, kde každý termín může nabývat pouze dvou hodnot:

- 1, pokud se termín v dokumentu vyskytuje
- 0 jinak

Kombinaci termínů v dotazu umožňují logické operátory `AND`, `OR`, `NOT`. Hlavní nevýhodou tohoto modelu je, že vzhledem k diskrétnímu oboru neumožňuje reprezentaci částečné shody [9]. Z tohoto důvodu se v praxi používá jen vzácně.

3.3.2 Vektorový model

Tento model je zřejmě nejrozšířenější. Dokument ve vektorovém prostoru je reprezentován jako vektor vah, které byly vypočteny na základě některé z metod vážení, z nichž dvě nejčastější budou nyní popsány.

Term frequency
Tato základní metoda vážení určuje důležitost slova podle toho, jak často se slovo nachází v daném dokumentu. Vydělení délkou dokumentu se provádí z toho důvodu, aby rozsáhlé dokumenty nebyly zvýhodněny. V takovém textu může být výskyt slova častější, nicméně to nemusí vypovídat o vztahu k obsahu.

$$tf_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}}, \tag{3.1}$$

kde $n_{i,j}$ je počet výskytů slova t_i v dokumentu d_j. Jmenovatel určíme jako součet výskytů slov v dokumentu d_j, tedy odpovídá jeho délce v počtu slov. Toto schéma nebere v úvahu, že jeden termín se může vyskytovat v mnoha dokumentech.

Term frequency - inverse document
Jedná se o způsob vážení termínů založený na frekvenci výskytu v textu, který se používá v mnoha úkolech zpracování přirozeného jazyka. Hodnota roste úměrně k četnosti výskytu slova v dokumentu a zároveň je kompenzována frekvencí výskytu slova v celém korpusu.

Zohledňuje se tedy skutečnost, že některá slova jsou běžnější než jiná a že se mohou vyskytovat ve více dokumentech. Jak je patrné už z názvu, tato metrika vychází z předchozího způsobu vážení, ale k výpočtu přidává druhou složku.

$$tf\_idf_{i,j} = tf_{i,j} \times idf_i \tag{3.2}$$

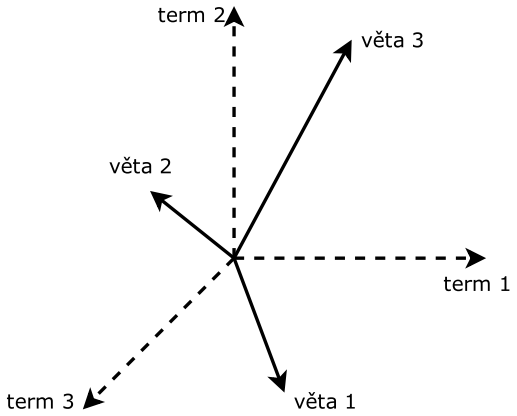
Idf složka reprezentuje selektivní sílu slova [15]. Čím častěji se slovo vyskytuje v dokumentech, tím menší selektivní sílu má. Např. určité předložky a spojky se vyskytují prakticky ve všech textech, navíc nemají žádný lexikální význam, tudíž pro reprezentaci dokumentu jsou bezvýznamné.

$$idf_i = \log \frac{|D|}{|\{j : t_i \in d_j\}|}, \tag{3.3}$$

kde $|D|$ je velikost korpusu dokumentu, tedy počet dokumentů, ve kterých hledáme a $|\{j : t_i \in d_j\}|$ je počet dokumentů, které obsahují slovo i.

Proč zavádíme vážení? Slova v textu mají různou důležitost pro význam obsahu věty nebo dokumentu. Vážení znamená přiřazení relativní hodnoty každému slovu, reprezentující důležitost jeho významu pro celkový kontext. Samotné váhy lze určit mnoha způsoby. Můžeme počítat výskyty jednotlivých termínů s tím, že ty frekventovanější jsou pro význam dokumentu důležitější. Další kritérium je umístění v textu, např. slova v názvu práce mají větší význam. Lze využít také mluvnický význam, jako jsou přiřazené slovní druhy [31].

Oproti předchozímu model vektorového prostoru nevrací binární odpověď, jestli je dokument relevantní dotazu. Na místo toho jsou dokumenty seřazeny podle stupně souvislosti a váha může nabývat jakékoliv hodnoty z intervalu $\langle 0, 1 \rangle$. Jak jsme si uvedli v sekci 2.3, v tomto modelu existuje mnoho metrik podobnosti. Jelikož jsou dokumenty a dotazy reprezentovány jako vektory, nejjednodušší je určit úhel mezi nimi. Standardem v tomto oboru je kosinová podobnost. Model vektorového prostoru je na obrázku 3.2.



Obrázek 3.2: Vektorový prostorový model.

3.3.3 Pravděpodobnostní model

Tento přístup je založen na statistice a pravděpodobnostech. Nejprve se vytvoří jazykový model pro každý dokument a poté jsou dokumenty ohodnoceny na základě pravděpodobnosti dotazu pro daný jazykový model. Nechť dotaz q je posloupnost termínů

$q = q_1, q_2, \cdots, q_m$ a D je množina dokumentů $d_1, d_2, \cdots, d_n$, pak pravděpodobnost, že dokument d_j generuje dotaz q je $Pr(q|d_j)$. Pro řazení dokumentů se pak používá pravděpodobnost $Pr(d_j|q)$, s použitím Bayesova pravidla dostáváme [8]:

$$Pr(d_j|q) = \frac{Pr(q|d_j)Pr(d_j)}{Pr(q)} \tag{3.4}$$

3.4 Distribuční sémantické modely

Distribuční sémantické modely (distributional semantic models, DSM) vychází z distribuční hypotézy, dle které se významově podobná slova vyskytují v podobných kontextech. Tuto hypotézu si blíže vysvětlíme na následujícím příkladu převzatém z [7]: Při čtení anglického textu jsme narazili na neznámé slovo *tezguino*. Bylo však použito v následujících větách:

A bottle of tezguino is on the table.
Everyone likes tezguino.
Tezguino makes you drunk.
We make tezguino out of corn.

I když je nám význam tohoto slova utajený, z uvedených vět můžeme usoudit, že se jedná o oblíbený alkoholický nápoj vyráběný z obilí (případně kukuřice). Dále si můžeme všimnout, že místo slova *tezguino* můžeme do věty dosadit např. *beer*, tedy *tezguino* a *beer* mají podobné významy. Distribuční hypotéza odpovídá principu, jakým se děti učí významu slov, s kterými se dosud nesetkaly.

DSM se vyznačují tím, že pro charakterizaci sémantiky slov je klíčová jejich statistická distribuce v kontextu. Reprezentují význam slov ve vektorech popisujících jejich vzorce spoluvýskytu s ostatními slovy. Různé úlohy, jako jsou identifikace analogických relací nebo klasifikace synonym, jsou poté vykonávány prostřednictvím operací lineární algebry s distribučními vektory. Dalším důvodem oblíbenosti DSM je, že překonávají jiné typy reprezentace, jako jsou sémantické sítě.

Distribuční hypotéza vede k jasnému způsobu porovnávání slov – vytvořit vysoce dimenzionální vektory prostřednictvím statistické analýzy výskytů a poté definovat sémantickou podobnost vektorů. Na základě DSM vznikly kompoziční DSM (compositional distributional semantic models, CDSM), reprezentující význam frází a vět skládáním distribučních reprezentací slov, které obsahují.

3.4.1 Latentní sémantické indexování

Latentní sémantické indexování (latent semantic indexing, LSI) je metoda extrakce a reprezentace kontextů slov pomoci statistického výpočtu aplikovaného na velký korpus dat. Hlavní ideou je, že shromáždění všech kontextů, ve kterých se dané slovo vyskytuje a ve kterých ne, poskytuje množinu omezení, která výrazně určuje podobnost významů slov mezi sebou. LSI se také používá k automatické kategorizaci dokumentů.

Během LSI je vytvořena matice, kde každá pozice obsahuje informace o vahách termů v daném dokumentu. Protože vytvářený slovník je založený na celé sbírce dokumentů, matice jsou rozsáhlé, a proto se dále dimenze matice redukuje. K tomuto účelu se používá singulární rozklad (singular value decomposition, SVD) [28]. Během snižování rozměru je část informace nenávratně ztracena, některé podobné vektory se sjednotí nebo jsou ve vektorovém prostoru promítnuty blíže k sobě.

Podobnost dokumentů je vypočtena jako kosinová podobnost nebo jinou vektorovou metrikou. LSI má kromě výpočetní složitosti i další omezení. Nezohledňuje pořadí slov ve větě, syntaktické vztahy ani morfologii. Dokáže extrahovat správnou reprezentaci textu i bez těchto vlastností, ovšem stále musíme zohledňovat, že výsledky jsou neúplné a mohou obsahovat chyby.

3.4.2 Hyperprostorová analogie jazyka

Hyperprostorová analogie jazyka (hyperspace analogue to language, HAL) modeluje sémantický prostor dle spoluvýskytu slov. Vytváří matici $N \times N$ elementů, kde N je počet slov ve slovníku, postupným pohybem po textu korpusu s čtecím rámcem n slov, typicky 10. Kdykoliv se do toho rámce dostanou libovolná dvě slova, na odpovídající pozici v matici dojde ke zvýšení asociace mezi nimi o jistou váženou hodnotu. Slova blíže zkoumanému slovu mají vyšší váhu, protože mohou obsahovat více sémantické informace související s analyzovaným slovem.

V HAL jsou dvě slova sémanticky související, pokud mají sklon vyskytovat se se stejnými slovy, tedy i v případě, že se na žádném místě nevyskytují spolu. Obvykle se k tvorbě sémantických vektorů využívají všechny informace o spoluvýskytu slov, ovšem ne všechny sloupce poskytují stejné množství informace k rozlišení významů slov. Některé nástroje proto počítají informační entropii každého sloupce a ponechají pouze ty, které se ukáží jako důležité [14]. HAL uchovává informaci o pořadí slov jednoduchým rozlišením, zda je přilehlé slov před nebo za zkoumaným slovem.

Kapitola 4

Nástroje

Tato kapitola se zaměřuje na získávání podobnosti slov z korpusu s využitím nástroje Word2Vec, který poskytuje efektivní implementace algoritmů k výpočtu vysoce dimenzionálních vektorových reprezentací slov na základě vstupu ve formě nestrukturovaného seznamu vět.Původně byl Word2Vec vytvořen výzkumnou skupinou Google[1] vedenou Tomášem Mikolovem.

V této práci jsem ale využil implementaci v jazyce python jako součást nástroje gensim [21]. Jedná se o open-source nástroj využívaný řadou společností. Je určen ke zpracování rozsáhlých korpusů, obsahuje implementace tf-idf, hlubokého učení, latentní sémantické analýzy a dalších algoritmů z oblasti NLP. Jiné systémy k určení podobnosti slov používají online lexikální systém WordNet.

4.1 Word2Vec

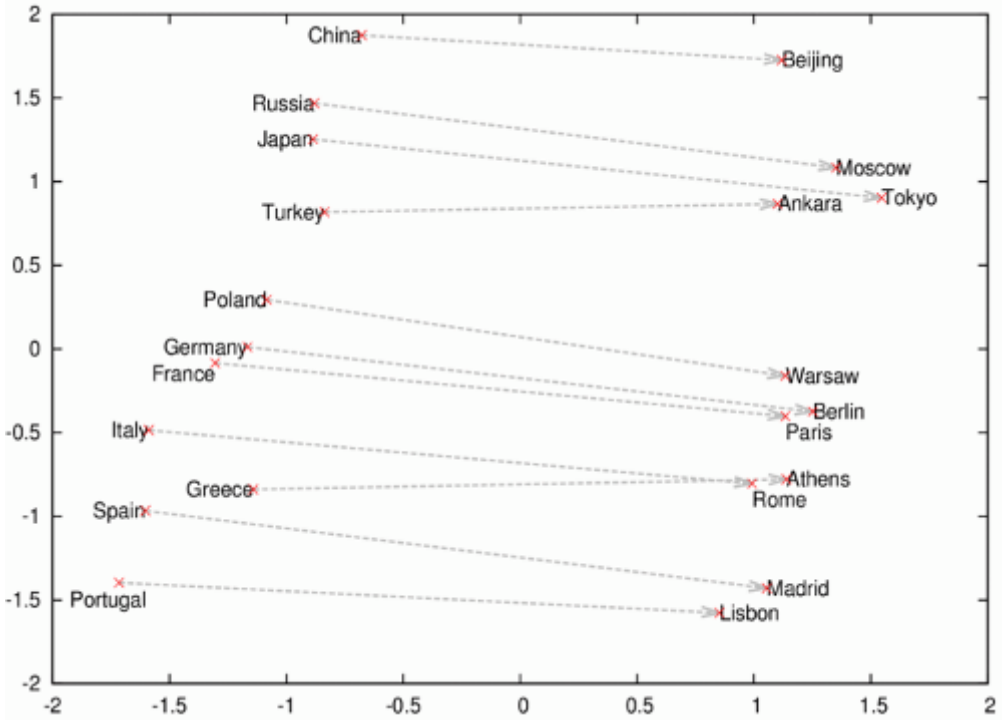
Jedná se o neuronovou síť transformující text do číslicové reprezentace, kterou jsou poté schopny zpracovat algoritmy hlubokého učení. Word2Vec vytváří rysy bez nutnosti lidského zásahu, včetně kontextu jednotlivých slov. Tento kontext je tvořen ve formě víceslovných čtecích okének. Pro dostatečné množství dat, kontextů a příkladů použití, Word2Vec může s vysokou pravděpodobností odhadovat význam slova na základě jeho výskytů.

Slova každé vstupní věty jsou transformovány na vektory a porovnány s ostatními vektorizovanými seznamy slov v n-dimenzionálním vektorovém prostoru. Související slova se v tomto prostoru nachází blízko sebe, podobně sousloví. Vektorizace slov umožňuje exaktně měřit jejich podobnost prostřednictvím kosinové podobnosti a shlukovat je.

Uvažujme např., že použijeme Word2Vec k vytvoření modelu vektorů slov vyskytujících se ve velkém korpusu novinových článků. Pokud výslednou reprezentaci vektorového prostoru měst a států promítneme do dvoudimenzionálního prostoru, můžeme pozorovat vzorce jako ty na obrázku 4.1. Nejenom podobné státy (např. geografickou polohou) jsou ve vektorovém prostoru blízko, ale vidíme také, že jejich hlavní města jsou rozmístěna v předvídatelných vzdálenostech, protože algoritmus hlubokého učení byl schopen zachytit představu vztahu hlavního města mezi dvěma entitami z nestrukturovaného textu.

Díky tomuto faktu, vektor získaný operací $vec(Paris) - vec(France) + vec(Germany)$ vyústí v pozici uvnitř modelu vektorového prostoru, která je blízko umístění slova *Berlín*. Zákonitosti ve vektorového reprezentaci mohou být využity k hledání slov souvisejících prostřednictvím určitého vztahu.

[1] Viz https://code.google.com/p/word2vec/.



Obrázek 4.1: Vektorový prostor měst a států promítnutý do 2D prostoru [18].

4.1.1 N-gram

Během trénování jsou slova čtena jako tzv. n-gramy. N-gram je sekvence n sousedících položek z dané slovní sekvence. Je to n-tá verze unigramu (jedna položka), bigramu (dvě položky), trigramu (tři položky). V literatuře se často používá také pojem kontextového okénka. V příkladu výše je kontextové okénko rovno 3, často se používá velikost 5. Tento n-gram je poslán do neuronové sítě k učení významu daného slova z jeho kontextu výskytu. Word2Vec používá různé druhy čtecích rámců: nepřerušovaný (continous) n-gram a skip-gram.

Jejich význam si ukážeme na následující větě: *I bought a new computer*. Ta může být rozdělena na 3 nepřerušované trigramy: (*I*, *bought*, *a*), (*bought*, *a*, *new*), (*a*, *new*, *computer*). Skip-gram naproti tomu umožňuje vynechávání slov. Z naší věty můžeme vytvořit řadu skip-gramů o délce tří slov: (*I*, *bought*, *new*), (*I*, *bought*, *computer*), (*I*, *a*, *new*) atd.

Předpočítání všech n-gramů umožňuje efektivní způsob vytváření vektorů slov na základě faktu, že seznam n-gramů zahrnuje všechna jednotlivá okénka pro každé slovo v korpusu spolu s jejich statistikou. Toto činí n-gram model komprimovanou reprezentací korpusu se všemi charakteristickými informacemi potřebnými k vytvoření distribučního sémantického modelu. Místo standardní techniky posouvání okénka přes korpus jako u HAL můžeme seskupit informace přímo z n-gramů.

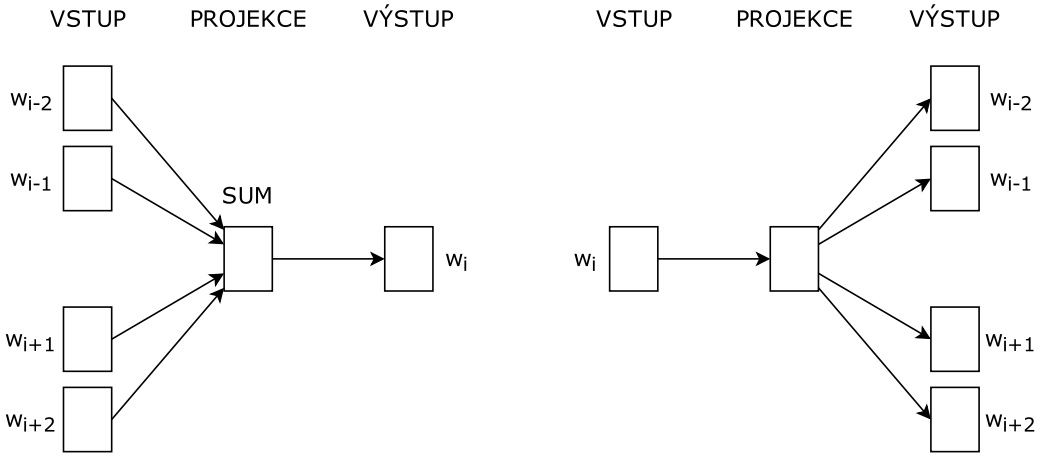
4.1.2 Continuous bag-of-words, skip-gram

Word2Vec poskytuje implementace dvou rozdílných algoritmů pro určení vektorů slov: continuous bag-of-words (CBOW, také bag-of-n-grams) a skip-gram (SG). Pro oba přístupy budeme předpokládat, že současné analyzované slovo ve větě je w_i. Vstupem CBOW archi-

tektury jsou předcházející a následující slova daného slova, např. w_{i-2}, w_{i-1}, w_{i+1}, w_{i+2}. Výstupem neuronové sítě pak bude w_i. Model tedy predikuje slovo podle daného kontextu, který může mít různou velikost.

Je populární zejména díky své jednoduchosti, efektivnosti a nečekané přesnosti. Má však také řadu nevýhod. Ztrácíme informaci o pořadí slov, a tak odlišné věty mohou mít naprosto stejnou reprezentaci, pokud obsahují stejná slova. Méně frekventovaná slova dosahují horší přesnosti.

Skip-gram pracuje obráceně. Vstupem tohoto modelu je slovo w_i, výstupem w_{i-2}, w_{i-1}, w_{i+1}, w_{i+2}. Jinými slovy, úkolem je odhadnout kontext na základě daného slova. Tento kontext na rozdíl od CBOW není omezen pouze bezprostředními slovy, trénované instance mohou být vytvořené vynecháním určitého počtu slov v jeho kontextu, např w_{i-4}, w_{i-3}, w_{i+3}, w_{i+4}. Velikost okénka určuje, jak dalece vpřed a zpět se bude algoritmus dívat k predikci slov. Činnost obou přístupů je znázorněná na obr. 4.2.



Obrázek 4.2: Znázornění architektur CBOW (vlevo) a Skip-gram (vpravo).

Např. výše použitá věta *I bought a new computer.* má tři trigramy, ovšem důležitým chybějícím trigramem vyplývajícím z věty může být také (*bought*, *a*, *computer*). Použití skip-gramů povoluje vynechání slova *new* a umožňuje tak vytvoření tohoto trigramu. Obecně SG pro danou vzdálenost vynechání k umožňují k nebo méně vynechání při konstrukci n-gramů.

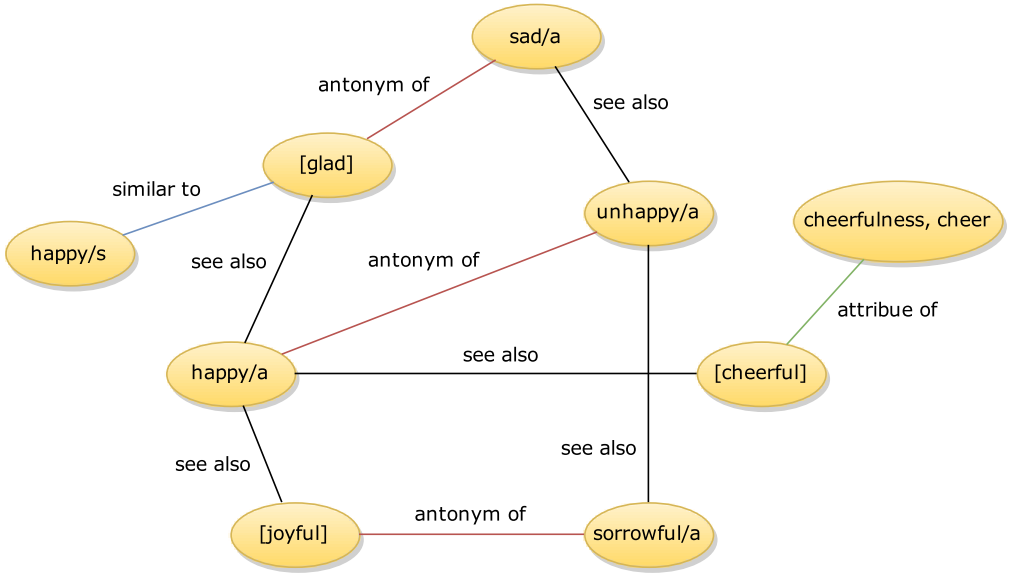
Reprezentace skip-gram se ukázala jako přesnější než předchozí model, protože generuje obecnější kontexty. Trénování CBOW je ovšem několikrát rychlejší, poskytuje vyšší přesnost u frekventovaných slov. SG dosahuje dalších zajímavých výsledků, protože jen prosté vektorové sčítání může často produkovat smysluplné výsledky. Např. $vec(Russia) + vec(river)$ je blízko $vec(Volga\ River)$. To ukazuje, že použitím základních matematických operací na vektorové reprezentace můžeme získat netriviální stupeň porozumění textu.

V této části jsem se inspiroval z [17], odkud je také původní obrázek.

4.2 WordNet

Jako alternativu k Word2Vec využívá mnoho systémů WordNet. Jedná se o lexikální systém vyvinutý na Princetonské univerzitě, který je volně dostupný, díky čemuž je stále jeden z nejpopulárnějších v oblasti zpracování přirozeného jazyka. Existuje i verze pro češtinu.

Jednotlivá slova jsou seskupena do tzv. synsetů, množin synonym. Synsety jsou organizovány podle různých typů vtahů vůči ostatním synsetům, vzniká tak hierarchická struktura, viz obrázek 4.3.



Obrázek 4.3: Znázornění struktury WordNet.

Tato struktura je poté důležitá při určování sémantické podobnosti $sim(w_1, w_2)$ slov w_1, w_2. Protože je báze znalostí hierarchická, můžeme se na ni dívat jako na graf. Pak podobnost mezi dvěma slovy lze určit jako délku nejkratší cesty mezi nimi.

$$sim(w_1, w_2) = len(w_1, w_2) \tag{4.1}$$

Jiné metody počítají hloubku uzlů ve struktuře $depth(w_i)$. Využívají k tomu tzv. lowest common subsumer, což je vlastně takový nadřazený uzel, který má nejkratší vzdálenost od porovnávaných pojmů. Např. *animal* a *mammal* jsou slova zahrnující *horse* a *cow*, ovšem jako lowest common subsumer určíme *mammal*, protože je to bližší a specifičtější nadřazené slovo.

$$sim(w_1, w_2) = depth(lcs(w_1, w_2)) \tag{4.2}$$

Nevýhodou tohoto nástroje je, že struktura musí být po celou dobu své existence udržována aktuální a tedy vyžaduje stálý dohled člověka. Jednotlivé verze se proto snaží reagovat na potřeby uživatelů a vývoj jazyka. Např. v starších verzích neexistovalo žádné spojení mezi slovy *tennis* a *forehand* nebo *serve*, ačkoliv většina lidí by se jistě shodla na tom, že se jedná o související pojmy. Verze 2.0 proto přišla se vztahem doménový pojem, který tyto volnější souvislosti zachytí. Po této změně tedy lze jak najít oblast, ve které se vyskytují slova *forehand* a *serve*, nebo jaké pojmy souvisí se zmíněnou doménou.

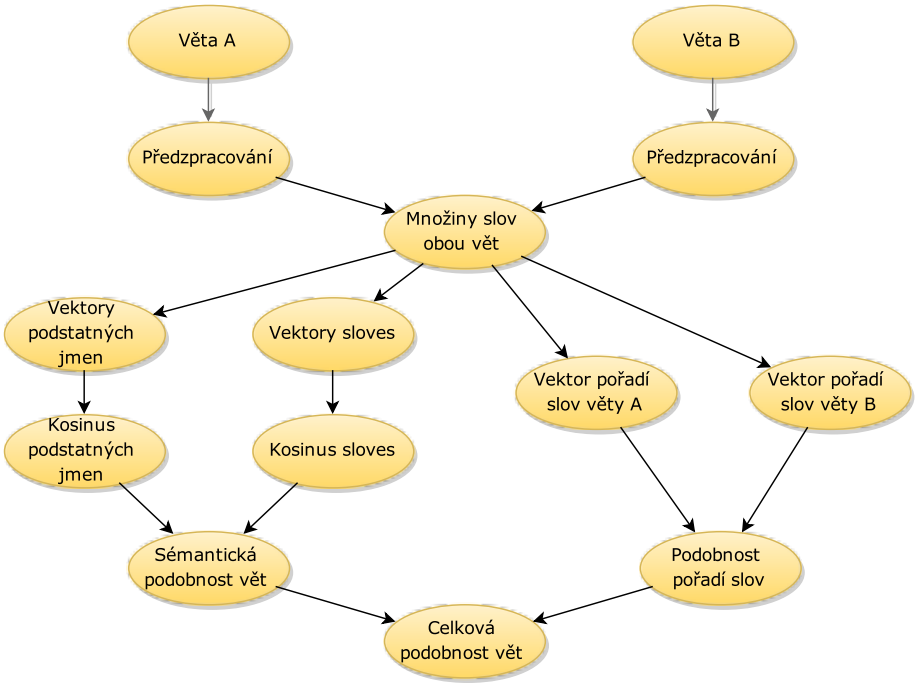
Další velký problém je, že synonyma jsou silně závislá na doméně. Některá slova jsou si podobná v jedné konkrétní oblasti, ale již méně v jiné, v závislosti jaké významy převažují v dané doméně.

Kapitola 5

Podobnost vět

V této kapitole využijeme výše popsané techniky určení podobnosti slov k vyhodnocení podobnosti vět, rozebereme také, v čem spočívá složitost tohoto problému a uvedeme možná řešení. Tradiční metody určování podobnosti delších textů se soustředí především na hledání shodných slov. Zatímco podobné dokumenty obvykle obsahují určitě množství stejných slov, v kratších textech, jako jsou věty, takový výskyt může být vzácný nebo se nemusí objevit vůbec, takže tento přístup pro porovnávání vět není vhodný. [13].

V tomto případě vyjadřuje hodnota podobnosti vztah mezi významy dvou vět. Jedná se o složitější problém než je tomu u slov, protože přesný význam věty závisí kromě kontextu také na pořadí slov. Metoda výpočtu podobnosti dvou vět uvedená dále v této sekci zohledňuje váhu a podobnost odlišných slovních druhů. Určuje výslednou podobnost z více částí, bere v úvahu i pořadí slov. Popis algoritmu znázorňuje obrázek 5.1. Hlavním zdrojem této sekce mi byl článek [11], obsahující také uvedenou metodu.



Obrázek 5.1: Diagram znázorňující algoritmus výpočtu celkové podobnosti významů dvou vět.

5.1 Sémantická podobnost vět

Věty se skládají ze slov různých druhů, klíčové je pochopit jejich funkcionalitu a důležitost pro význam věty. Podstatná jména obvykle slouží jako podmět nebo předmět věty, slovesa pak jako přísudek. Společně nesou největší sémantický význam, ostatní druhy nejsou z hlediska významu tolik důležité. Z tohoto důvodu se upravuje předzpracování dokumentů, POS tagging je nezbytnou součástí algoritmu. Po předzpracování odfiltrujeme některá slova nedůležitá v této metodě, pro každou větu vytvoříme množinu sloves a množinu podstatných jmen.

V definici 1 SEN_A a SEN_B jsou množiny slov po předzpracování, S_VA a S_NA jsou množiny sloves a podstatných jmen SEN_A. NV_{SEN_A} a VV_{SEN_A} jsou prostory sloves a jmen ve větě A, obdobně pro větu B. Dle definice 2 sémantický prostor podstatných jmen a sémantický prostor sloves je definován jako sjednocení jmen v SEN_A a SEN_B, respektive sjednocení sloves v SEN_A a SEN_B.

Definice 1. Množiny slov vět SEN_A a SEN_B.
$SEN_A = \{S_{VA}, S_{NA}\}$
$SEN_B = \{S_{VB}, S_{NB}\}$

Definice 2. Vektor podstatných jmen (NV), vektor sloves (VV), sémantický prostor podstatných jmen, sémantický prostor sloves.
Vektor podstatných jmen odpovídá prostoru ($S_{NA} \cup S_{NB}$), vektor sloves odpovídá prostoru ($S_{VA} \cup S_{VB}$), dále
$|NV_{SEN\_A}| = |NV_{SEN\_B}| = |S_{NA} \cup S_{NB}|$
$|VV_{SEN\_A}| = |VV_{SEN\_B}| = |S_{VA} \cup S_{VB}|$

Pro každé slovo z NV nebo VV je spočítána podobnost vůči celému odpovídajícímu sémantickému prostoru a největší hodnota je zvolena jako konečná hodnota každého pole v tabulce. Formálně zapsáno takto:

$$NV_{SEN\_A_i} = \max_{k=1}^{|S_{VA} \cup S_{VB}|} [Similarity(WORD_A, NOUN_k)] \tag{5.1}$$

$$VV_{SEN\_A_i} = \max_{k=1}^{|S_{VA} \cup S_{VB}|} [Similarity(WORD_A, VERB_k)] \tag{5.2}$$

Poté se vypočítá kosinová podobnost VV a NV vět, tedy kosinus sloves a kosinus podstatných jmen.

$$NC_{A,B} = \left(\frac{NV_{SEN\_A} \cdot NV_{SEN\_B}}{||NV_{SEN\_A}||\ ||NV_{SEN\_B}||} \right)^2 = \left(\frac{\sum_{i=1}^{|S_{VA} \cup S_{VB}|} NV_{SEN\_A_i} \times NV_{SEN\_B_i}}{\sqrt{NV^2_{SEN\_A_i}} \times \sqrt{NV^2_{SEN\_B_i}}} \right)^2 \tag{5.3}$$

$$VC_{A,B} = \left(\frac{VV_{SEN\_A} \cdot VV_{SEN\_B}}{||VV_{SEN\_A}||\ ||VV_{SEN\_B}||} \right)^2 = \left(\frac{\sum_{i=1}^{|S_{NA} \cup S_{NB}|} VV_{SEN\_A_i} \times VV_{SEN\_B_i}}{\sqrt{VV^2_{SEN\_A_i}} \times \sqrt{VV^2_{SEN\_B_i}}} \right)^2 \tag{5.4}$$

Na závěr se zkombinují hodnoty VC a NC do jednoho skóre. Váhy VC a NC jsou přizpůsobeny tzv. balančním koeficientem ξ, jehož hodnotu nejprve odhadneme a poté doladíme experimenty. Podstatná jména mají přibližně stejný vliv na význam věty jako slovesa, ovšem jejich počet ve větě obyčejně bývá vyšší. Proto se volí hodnota ξ větší než $0,5$.

$$Similarity_{A,B} = \xi \times (NC_{A,B}) + (1 - \xi) \times (VC_{A,B}) \tag{5.5}$$

Faktu, že různé slovní druhy mají odlišný vliv na význam věty, využívají i další způsoby. Slova porovnávané věty můžeme např. rozdělit do jiných dvou množin. První bude obsahovat veškerá podstatná jména a slovesa, druhá zbylé slovní druhy. Ukázalo se však, že tato druhá množina má na podobnost vět jen minimální vliv a tomu odpovídal balanční koeficient. V této subsekci jsem využil podrobně popsanou metodu výpočtu sémantické podobnosti vět uvedenou v [10].

Příklad 1.
Nyní si na příkladu ukážeme činnost uvedeného algoritmu. Pro níže uvedené věty uvádím v závorkách slovní druhy všech slov podle notace v tabulce. Jako balanční koeficient použijeme hodnotu $\xi = 0,65$.

Věta A: *Food*[N] *is*[V] *also*[S] *included*[V] *in*[S] *the*[S] *price*[N] *of*[C] *the*[C] *accommodation*[N].
Věta B: *The*[C] *price*[N] *of*[C] *the*[C] *accommodation*[N] *also*[C] *includes*[V] *the*[C] *food*[C].
Věta C: *He*[N] *introduced*[V] *better*[J] *methods*[N] *of*[C] *management*[N] *in*[C] *this*[C] *company*[N].

V	Sloveso
N	Podstatné jméno
J	Přídavné jméno
A	Příslovce
S	Stop word

Tabulka 5.1: Význam notace.

$NV_{SEN_A}/A-B$ prostor jmen	food	price	accomodation
food	1	0,43	0,59
price	0,43	1	0,59
accomodation	0,59	0,59	1
Vector	1	1	1

Tabulka 5.2: $NV_{SEN_A}/A-B$ a prostor podstatných jmen.

Nejprve budeme porovnávat dvojici A, B. Po předzpracování získáme pro větu A množinu podstatných jmen $S_{NA} = \{food, price, accommodation\}$ a sloves $S_{VA} = \{be, include\}$, obdobně pro větu B $S_{NB} = \{price, accommodation, food\}$, $S_{VB} = \{include\}$. Sémantický prostor podstatných jmen je pak $\{food, price, accommodation\}$, sloves $\{be, include\}$. Vektory jmen a sloves obou vět jsou:

$NV_{SEN\_A} = (1, 1, 1) = NV_{SEN\_B}$

$VV_{SEN_A}/A-B$ prostor sloves	be	include
be	1	0
include	0	1
Vector	1	1

Tabulka 5.3: $VV_{SEN_A}/A-B$ a prostor sloves.

$VV_{SEN\_A} = (1, 1)$
$VV_{SEN\_B} = (0, 1)$

Dalším krokem je výpočet kosinové podobnosti.

$$NC_{A,B} = \left(\frac{NV_{SEN\_A} \cdot NV_{SEN\_B}}{||NV_{SEN\_A}||\ ||NV_{SEN\_B}||}\right)^2 = \left(\frac{1 \times 1 + 1 \times 1 + 1 \times 1}{\sqrt{1^2 + 1^2 + 1^2} \times \sqrt{1^2 + 1^2 + 1^2}}\right)^2 = 1$$

$$VC_{A,B} = \left(\frac{VV_{SEN\_A} \cdot VV_{SEN\_B}}{||VV_{SEN\_A}||\ ||VV_{SEN\_B}||}\right)^2 = \left(\frac{1 \times 0 + 1 \times 1}{\sqrt{1^2 + 1^2} \times \sqrt{0^2 + 1^2}}\right)^2 = 0,666$$

Výslednou podobnost vět A a B určíme takto:

$$Similarity_{A,B} = \xi \times (NC_{A,B}) + (1 - \xi) \times (VC_{A,B}) = 0,65 \times 1 + 0,35 \times 0,666 = 0,8831$$

Pokud bychom vypočítali i podobnost mezi A a C, vyšlo by nám $Similarity_{A,C} = 0,568$ Z těchto hodnot vyplývá, že větě A je více podobná věta B než C, což odpovídá lidskému vnímání významu.

5.2 Podobnost pořadí slov

Podobnost pořadí slov, někdy označovaná také jako syntaktická nebo strukturní podobnost, poskytuje informaci o pozičních vztazích mezi slovy – která slova se vyskytují ve větě, která jim předchází nebo je následují.

Uvažme dvojici vět S_1 a S_2, které obsahují přesně ta samá slova ve stejném pořadí, kromě dvou slov z S_1, která se v S_2 vyskytují v pořadí opačném:

$S_1 =$ *A strong man is hunting the old lion.*
$S_2 =$ *A strong lion is hunting the old man.*

Jelikož se věty skládají ze stejných slov, jakákoliv metoda založená na Bag of Words určí, že se jedná o věty totožné. Avšak jak víme, věty obsahující stejná slova, ovšem v odlišném pořadí, mohou vést k odlišným významům. Stejně tak je tomu v tomto případě. Pro člověka je snadné odlišit význam na základě pořadí slov, ovšem náš algoritmus toho musí být schopen také.

Použitou metodu určení podobnosti pořadí slov si vysvětlíme na výše uvedených větách. Průnik jejich slov je množina $S = \{a, strong, man, is, hunting, the, old, lion\}$. Každému slovu v S_1 a S_2 přiřadíme unikátní index odpovídající pořadí, v jakém se slova ve větě

nachází. Např. index 3 v první větě značí slovo *man*. Dále vektor pořadí slov r je vytvořen pro obě věty na základě průniku slov obou vět S. Chceme-li tento vektor r_1 vytvořit např. pro větu S_1, tak pro každé slovo w_i z S se pokusíme najít stejné nebo nejpodobnější slovo v S_1 tímto postupem:

1. Pokud věta obsahuje stejné slovo, vyplníme záznam pro toto slovo ve vektoru r_1 odpovídajícím indexem z S_1. Jinak se pokusíme najít nejpodobnější slovo $\tilde{w}_i$.

2. Pokud je podobnost mezi w_i a $\tilde{w}_i$ větší než zvolený práh, záznam w_i v r_1 vyplníme indexem tohoto nejpodobnějšího slova $\tilde{w}_i$ z S_1.

3. Pokud předchozí hledání selže, záznam w_i v r_1 doplníme nulou.

Když tuto metodu aplikujeme na zmíněné věty, získané vektory pořadí slov mají tvar:

$r_1 = (1, 2, 3, 4, 5, 6, 7, 8)$
$r_2 = (1, 2, 8, 4, 5, 6, 7, 3)$

Vektory reprezentují strukturní informaci obsaženou ve větě a použijeme je k výpočtu podobnosti pořadí $Similarity_{str}$:

$$Similarity_{str} = 1 - \frac{||r_1 - r_2||}{||r_1 + r_2||} \tag{5.6}$$

5.3 Celková podobnost vět

Sémantická podobnost reprezentuje lexikální blízkost, syntaktická podobnost zase nese informaci o stavbě vět. V některých případech bychom si vystačili s kteroukoli metrikou použitou samostatně, ve spoustě dalších by ovšem tento přístup selhal. Díky jejich kombinaci se mezivýsledky vhodně doplňují a odstraňují vzájemné nedostatky. Známe-li tedy sémantickou podobnost $Similarity_{sem}$ a strukturní podobnost $Similarity_{str}$ daných dvou vět, vypočítáme jejich celkovou podobnost $Similarity_{sen}$ jako:

$$Similarity_{sen} = \omega \times (Similarity_{sem}) + (1 - \omega) \times (Similarity_{str}), \tag{5.7}$$

kde ω je váhový koeficient každé části. Počáteční hodnotu parametru se doporučuje nastavit v intervalu $\langle 0, 5, 1 \rangle$, protože syntaxe hraje v podobnosti vět podřadnou roli. Přesná hodnota závisí na konkrétním jazyku a úspěšnosti předchozích kroků, proto je třeba provést další testování.

5.4 SemEval 2014, úkol 1

SemEval[1] (Semantic Evaluation) tvoří nástroje k identifikaci a řešení problémů souvisejících s významem řeči, jeho předchůdcem je Senseval. Začal s jednoduchými pokusy výpočtu významu slova, přes zkoumání vzájemných souvislostí mezi větnými členy se dostal k problémům odhalování vztahů mezi větami a povahou obsahu řeči, kterými se zabývá dnes. Každoročně SemEval vypisuje několik úkolů souvisejících s významem jazyka, k jejichž vypracování se mohou přihlásit dobrovolní účastníci.

[1] Viz na http://alt.qcri.org/semeval2015/.

Každý úkol obsahuje přesné zadání, příklady či trénovací data a popis vyhodnocení. Způsob řešení problému není nijak omezen, odevzdává se typicky jen výstup systému a jeho popis. Po skončení termínu jsou zveřejněny výsledky odevzdaných řešení. Výstupy se poté analyzují na konferenci, diskutují se použité přístupy a je zveřejněna zpráva o úkolu. Jako součást teto práce a abych se blíže seznámil s nástrojem Word2Vec, jeden takový úkol jsem se pokusil vypracovat. Následující části proto popisují zadání úkolu a jeho řešení.

5.4.1 Zadání úkolu

Úkol[2] se skládal ze dvou částí:

1. souvislost vět - určení stupně sémantické podobnosti mezi dvěma větami
2. vyplývání vět - detekce vyvozování významu mezi větami

V první části bylo cílem pro dané dvě věty A, B určit skóre, které značí, jak moc příbuzný význam mají. Skóre tentokrát nabývá hodnot z intervalu $\langle 0, 5 \rangle$, kde 5 značí plnou shodu. Tabulka 5.4 ukazuje příklady dvojic vět s různými stupni sémantické souvislosti.

Skóre souvislosti	**Vstupní věty**
1.6	A: *A man is jumping into an empty pool.* B: *There is no biker jumping in the air.*
2.9	A: *Two children are lying in the snow and are making snow angels.* B: *Two angels are making snow on the lying children.*
4.9	A: *A person in a black jacket is doing tricks on a motorbike.* B: *A man in a black jacket is doing tricks on a motorbike.*

Tabulka 5.4: Příklady dvojic vět s odpovídajícím skóre jejich souvislosti.

V druhé části úkolu se pro stejnou dvojici vět snažíme určit, jak význam B vyplývá z A. Systém tedy každé dvojici vět přiřadí značku ENTAILMENT, když z A vyplývá B, CONTRADICTION, pokud A popírá význam B, nebo NEUTRAL, když pravdivost B nemůžeme z A nijak odhadnout. Tabulka 5.5 ukazuje příklady dvojic vět s různými vztahy vyplývání.

Skóre souvislosti	**Vstupní věty**
ENTAILMENT	A: *Two teams are competing in a football match.* B: *Two groups of people are playing football.*
CONTRADICTION	A: *The brown horse is near a red barrel at the rodeo.* B: *The brown horse is far from a red barrel at the rodeo.*
NEUTRAL	A: *A man in a black jacket is doing tricks on a motorbike.* B: *A person is riding the bicycle on one wheel.*

Tabulka 5.5: Příklady dvojic vět s odpovídajícím přiřazením jejich vyplývání.

Obě části úkolu byly vyhodnocení použitím standardních metrik. Výstup první části byl ohodnocen Pearsonovou korelací, která byla vybrána jako oficiální měřítko k určení pořadí účastníků, dále pak Spearmanovou korelací a střední kvadratickou odchylku, výsledky druhé části vyjadřovala přesnost.

[2]Přesné zadání lze nalézt na http://alt.qcri.org/semeval2014/task1/.

5.4.2 Datová sada SICK

K testování systémů účastníků soutěže byla použita datová sada SICK (Sentences Involving Compositional Knowledge), obsahující dvojice anglických vět anotovaných pro potřeby úkolu. Jednotlivé věty v dvojici se často liší použitím blízké parafráze, opaku či velice protikladného údaje, nebo úplně nesouvisejícího obsahu. Ve většině případů se jedná o souvětí skládající se ze dvou vět.

5.4.3 Přístupy účastníků

Přístup k řešení problému nebyl nijak omezen. Úkolu se nakonec zúčastnilo 21 týmů, z nichž většina soutěžila v obou jeho částech. Díky datové sadě SICK vypracování úkolu nevyžaduje příliš externích nástrojů, jako jsou ontologie nebo systémy k rozpoznávání jmenných entit. Většina účastníků použila pouze standardní nástroje předzpracování přirozeného jazyka (tokenizátory, POS taggery) a relativně málo dalších zdrojů znalostí (WordNet, další korpus, vektorový sémantický model, databázi parafrází, systémy k získávání nejrůznějších syntaktických znaků).

Téměř všechny systémy kombinují několik druhů přístupů. Z tohoto důvodu rozdělíme systémy na kompoziční, tedy takové, které kompozičně vypočtou význam celé věty, avšak ne nezbytně přiřazením významů syntaktickým částem, a na nekompoziční. Celkem 13 systémů využilo kompozici alespoň v jednom z úkolů, 10 použilo skládání pro celé věty a 6 jenom pro fráze. Důležité je, že nejúspěšnější systémy patřily právě mezi těchto 13.

Dále analýzy výsledků ukázaly, že plně kompoziční systémy selhávají při zachycení kontradikcí. Jsou lepší než částečně kompoziční modely založené na frázích v odhalování kladného vyplývání, ale horší v určování neutrálních dvojic. Systémy využívající distribučního přístupu používaly několik typů DSM, od vektorových prostorových modely, přes modely témat, až po neuronové jazykové modely.

Řada účastníků záměrně extrahovala rysy, které přímo nenapomáhaly porozumění významu věty, ale poskytovaly určitou užitečnou charakteristiku SICK dat. Např. pro druhou část tohoto úkolu bylo dokázáno, že silně závisí na antonymech a slovech s negativním významem. Jeden z účastněných týmů dokonce zaznamenal, že pouze zjišťováním přítomnosti záporných slov může být detekováno 86,4 % párů kontradikcí a kombinováním překrývání slov a antonym lze určit 82,6 % dvojic vyplývajících vět a 83,6 % neutrálních dvojic.

5.4.4 Řešení

Mým cílem bylo pokusit se co nejlépe vyřešit obě části daného úkolu pouze s pomocí Word2Vec, bez použití dalších nástrojů, jako jsou taggery či stemmery. Jako model pro tuto úlohu jsem využil reprezentaci publikovanou společností Google[3], která byla trénována na asi 100 miliardách slov z novinových článků. Model obsahuje 300 dimenzionální vektory pro 3 miliony slov a frází. Jedná se o komplexní prostředek dosahující vysoké přesnosti, ačkoliv data nebyla příliš předzpracována.

Načtená dvojice vět je nejprve jednoduše předzpracována, což zahrnuje tokenizaci, odstranění členů, předložek a jiných nedůležitých prvků vět. Výhodou Word2Vec je, že nevyžaduje lematizaci vstupu, neboť odvozená slova mají již z principu vysokou podobnost. Základem řešení první části úkolu je funkce Word2Vec *n_similarity()*, která určuje kosinovou podobnost pro dvě množiny slov. V případě, kdy mají věty několik souvisejících slov

[3]Dostupná na adrese https://code.google.com/p/word2vec/.

a také skupinu slov zcela odlišnou, je vrácená podobnost zpravidla vyšší než vyžaduje daný úkol.

Z tohoto důvodu provádím rozklad každé věty na dvě podmnožiny slov podle toho, zda existuje k danému slovu v druhé větě slovo významově podobné. Kosinová podobnost je vypočítána pro obě dvojice podmnožin zvlášť a do výsledné podobnosti se oba dílčí výsledky započítají s vahami na základě velikosti obou skupin množin. Vhodným vážením snadno dosáhneme toto, že malý počet nesouvisejících slov je prakticky ignorován, zatímco u vět s odlišnými částmi tato druhá složka provádí jakousi penalizaci.

Předchozí postup je založený na Bag of Words, je tedy nutné zohlednit pořadí slov. K tomuto účelu používám metodu uvedenou v sekci 5.2, porovnávající pořadí podobných slov. Tento jednoduchý algoritmus zcela vynechávám, pokud se nepodaří najít dostatečné množství odpovídajících si slov mezi větami, v opačném případě je podobnost pořadí slov zprůměrována se sémantickou podobnosti s vahami dle počtu těchto dvojic.

Váha pořadí bývá zpravidla malá, menší než polovina, protože syntaktická podobnost nehraje při určování takovou roli a metoda je také pro řadu případů nepřesná. Uvážíme-li např. věty *This group of people is practicing water safety and wearing prepare.*, *Prepare are being worn by a group of people who are practicing water safety.*, jejich význam je prakticky totožný, což také vyjadřuje podobnostní skóre v trénovací sadě 4.9, ovšem jak vidíme, pořadí slov obou vět si příliš neodpovídá.

Druhé části úkolu jsem již nevěnoval tolik úsilí, protože přesná detekce vztahů mezi větami by kromě Word2Vec vyžadovala použití dalších nástrojů. Hlavními rozhodovacími prvky byly skóre podobnosti z první části a společná slova obou vět. Pro identifikaci vět s opačným významem jsem v druhé části úkolu využíval podobně jako mnoho jiných systémů extrakci slov s negativním významem.

5.4.5 Vyhodnocení

Jak je patrné z tabulky 5.6, tento program by se umístil v první části úkolu na 9. místě ze 17 účastníků s úspěšností 73 % a odchylkou 0,42, která však byla výrazně menší než byl průměr (0,63). V druhé části úkolu bych pak kvůli přesnosti 71 % patřil mezi horší týmy. Je třeba zdůraznit, že lepším výsledkům v obou částech brání nepoužití dalších nástrojů a omezení samotného Word2Vec. Ten se totiž k tomuto úkolů příliš nehodí, což demonstrují následující 4 typy chyb.

Nejlepší Pearsonovo skóre	82 %
Průměrné Pearsonovo skóre	72 %
Mé dosažené Pearsonovo skóre	73 %
Nejlepší úspěšnost vyplývání vět	84 %
Průměrná úspěšnost vyplývání vět	75 %
Mnou dosažená úspěšnost vyplývání vět	71 %

Tabulka 5.6: Porovnání dosažených výsledků s ostatními účastníky.

Věty v první skupině jsou si velice podobné, liší se zpravidla v několika málo slovech, která jsou buď přímo antonyma (*without, with*), nebo mají výrazně kontrastní význam (*attacking, helping*). Word2Vec ovšem určuje antonyma jako slova související, proto dostáváme odlišné hodnoty. Pro tuto první skupinu vět je typické, že podobnost vrácená mým systémem je vyšší než požadované skóre, viz tabulka. Jedinou možností, jak tyto chyby odstranit, je použití dalšího nástroje.

Příklady těchto chyb jsou uveden v tabulce 5.7, symbol $\surd$ označuje správnou hodnotu, $\times$ zase chybný výsledek mého řešeni.

Vstupní věty	$\surd$	$\times$
A: *A brown dog is attacking another animal in front of the man in pants.* B: *A brown dog is helping another animal in front of the man in pants.*	3, 7	4, 7
A: *A little girl is looking at a woman in costume.* B: *The little girl is looking at a man in costume.*	3, 8	4, 9
A: *Few people are eating at red tables in a restaurant without lights.* B: *Various people are eating at red tables in a crowded restaurant with purple lights.*	3, 2	4, 7

Tabulka 5.7: Příklady první skupiny chyb.

Druhá skupina chyb souvisí s určováním podobnosti pořadí slov. Věty v dvojici jsou složeny z téměř totožných slov, ale jemně se liší jejich pořadím. Většinou se jedná o prohození předmětů, přídavných jmen nebo předložek, které má na celkový význam silný vliv. Ve vzácných případech problémy způsobuje i přidáním/odebráním slova již obsaženého ve větě. Výsledek výpočtu podobnosti pořadí slov sice tyto změny reflektuje, ale již dostatečně nezpřesní zpravidla vysoké hodnoty BOW. I v tomto případě můj program vrací vyšší hodnoty, viz 5.8.

Zkoušel jsem tento problém řešit použitím taggeru a vážením jednotlivých slov v měření syntaktické podobnosti podle slovního druhu, ale ukázalo se, že stejně velkou změnu významu můžou způsobit přehozená podstatná jména i předložky.

Vstupní věty	$\surd$	$\times$
A: *A person in a black jacket is doing tricks on a motorbike.* B: *A person on a black motorbike is doing tricks with a jacket.*	3, 0	4, 1
A: *A little girl is looking at a woman in costume.* B: *A little girl in costume looks like a woman.*	2, 9	4, 1
A: *A dog is chasing another and is holding a stick in its mouth* B: *A dog is chasing a stick and holding another dog in its mouth.*	3, 2	4, 4

Tabulka 5.8: Příklady druhé skupiny chyb.

V další skupině se jedná prakticky o opačný problém, protože podobnost pořadí slov tentokrát způsobuje odchylku od správného výsledku, což společně s předchozí skupinou znamená, že tyto chyby nemohou být řešeny změnou váhy sémantické podobnosti. Tentokrát se pořadí slov liší výrazně, avšak významy obou vět ve dvojici jsou blízké. Způsobuje to použití trpného rodu, záměna přívlastku vlastního a nevlastního, či slova ze skupiny *that*, *who*, *what*.

Často je slovo jedné věty popsáno skupinou slov odlišných pro dvojici vět. V takovém případě se kromě podobnosti pořadí slov snižuje podobnost BOW, ve výsledku pak dostáváme nižší hodnoty celkové podobnosti než jsou požadované. Odchylka u těchto případů je ovšem nejmenší. Příklady chyb z této skupiny jsou v tabulce 5.9.

Poslední skupina chyb již nemá zjevnou společnou příčinu. Do jisté míry se zde mohou projevovat předchozí problémy (*standing, sitting*), specifické pojmy, věty extrémních rozměrů, ale jinak na tyto chyby můžeme pohlížet jako na nedostatky samotného programu. Jak již bylo řečeno, určení podobnosti je do jisté míry subjektivní problém a tudíž můžeme

Vstupní věty	√	×
A: *This group of people is practicing water safety and wearing preservers.* B: *Preservers are being worn by a group of people who are practicing water safety.*	4, 9	4, 1
A: *The player is dunking the basketball into the net and a crowd is in background.* B: *The ball is being dunked by a man with a jersey at a basketball game.*	4, 0	3, 4
A: *Two teams are competing in a football match.* B: *Football is being played by two competing teams.*	4, 8	4, 3

Tabulka 5.9: Příklady třetí skupiny chyb.

diskutovat nad odchylkami výsledků. Např. první dvojice vět uvedená v tabulce 5.10 by dle někoho mohla dosahovat vyšších hodnot než vzorových, jelikož indiánská čelenka (*Indian headdress*) jistě může blízce souviset s černou čelenkou.

Vstupní věty	√	×
A: *A woman is wearing an Indian headdress.* B: *The woman is wearing glasses and a black headdress.*	3, 0	4, 2
A: *A man a woman and two girls are walking on the beach.* B: *A group of people is near the sea.*	3, 8	2, 6
A: *A boy is standing in the water* B: *The boy is sitting near the blue ocean.*	2, 8	3, 8

Tabulka 5.10: Příklady čtvrté skupiny chyb.

Na řešení tohoto úkolu jsem zkoušel aplikovat také nástroj Doc2Vec, obdobu Word2Vec pro delší texty, avšak ukázalo se, že výsledky se od požadovaných značně liší. Podobnost významů dvou vět byla silně závislá na počtu společných slov, tudíž v případě, kdy jedna věta obsahuje nějaké další informace, je výstup znehodnocen. V některých systémech by ovšem Doc2Vec jistě našlo uplatnění.

Kapitola 6

Zpracování biomedicínských dat

Korpusy extrémních velikostí mohou způsobovat problémy dokonce i plně automatickému zpracovávání, proto se data rozdělují na menší celky a vytváří se tak modely pro jednotlivé domény. Tímto způsobem lze také snížit odchylku způsobenou mnohoznačností slov z různých oblastí. Většina snahy domény se tedy zaměřuje jenom na malou podmnožinu literatury. Aby došlo k prospěchu širší společnosti a zároveň se předešlo opakování činnosti, je žádoucí shromáždit a distribuovat standardní datové sady a z nich odvezené reprezentace.

V této kapitole ukážeme proces vytváření nového jazykové prostředku, vytvořeného z dostupné biomedicínské vědecké literatury. Tento proces sestává z následujících kroků:

- získání dostupného biomedicínského korpusu
- zpracování korpusu do vhodné podoby
- získání reprezentace vektorového prostoru prostřednictvím trénování Word2Vec
- porovnání výsledků a posouzení, jak dobře model zachycuje sémantické vzdálenosti a vztahy

6.1 Popis dat

Word2Vec, představený v kapitole 4, demonstroval metody, které umožňují určení slovní reprezentace z korpusu miliard slov. Např. by Word2Vec mohlo pomáhat při správě strukturovaných znalostních bází a ontologií nebo při zlepšování algoritmů získávání informací. Použitelnost těchto metod na reálná, oborově specifická data, vyžaduje hlubší zhodnocení.

Navzdory důležitosti takových přístupů ke zpracování medicínského jazyka, jen výjimečně se objevilo úsilí aplikovat je speciálně na biomedicínskou literaturu. Většinou se však jednalo spíše o teoretické práce, jako např. [6]. Algoritmy hlubokého učení jako Word2Vec jsou známé vyžadováním velmi velkých objemů trénovacích dat k získání dobrých výsledků, zatímco objem přístupné, vysoce kvalitní literatury specifického oboru, jako je medicína, byl často omezený. To může způsobovat zhoršení výsledků.

I přes úsilí vytvořit anotované zdroje pro nejrůznější úkoly NLP biomedicínských dat, počet neanotovaných doménových dokumentů vyčnívá oproti anotovaným o několik řádů. Literatura MEDLINE databáze z National Library of Medicine[1] (NLM) poskytuje přístup k více než 23 milionům citací z medicíny, biochemie, genetiky, embryologie a dalších oborů,

[1] Dostupné na http://www.nlm.nih.gov/.

efektivně pokrývajících celý rozsah biomedicíny. Představuje neanotovaný korpus v řádu miliard tokenů, tvořící reprezentativní korpus dané domény.

K další práci jsem využil distribuci MEDLINE dat konkrétně pro rok 2015. Jedná se o 779 souborů, které jsou systematicky tříděny dle roku publikace a obsahují data v jednom z následujících formátů – MEDLINE, PubMed-not-MEDLINE, nebo OLDMEDLINE MedlineCitation. Až na poslední soubor obsahuje každý soubor přesně 30 000 záznamů. Charakteristika databáze je uvedena v tabulce 6.1.

Počet souborů	779
Komprimovaná velikost	15, 8 GB
Nekomprimovaná velikost	113, 9 GB
Velikost čistého textu	15, 9 GB

Tabulka 6.1: Charakteristika dat získaných z databáze MEDLINE.

Celkem je v databázi uloženo 23 343 329 záznamů s popisem publikovaných článků, jako je jeho identifikátor, titulek nebo jméno časopisu, ve kterém byl publikován. 11 514 344 položek obsahuje rovněž abstrakt článku, což tvoří největší část čistého textu. Ostatní uložené informace nejsou z hlediska této práce tolik důležité. Hlavním jazykem korpusu je angličtina, ale jak je typické pro medicínu, objevují se zde i latinské výrazy.

Text získaný z uvedeného zdroje vyžaduje specifické předzpracování než bude využit k trénování vektorového modelu, protože Word2Vec neobsahuje žádné vestavěné funkce pro tuto činnost. Podobně jako běžné texty i biomedicínská data obsahují množství syntaktických odchylek a interpunkce, což má negativní vliv na činnost algoritmu a kvalitu výsledného modelu vektorového prostoru. Vlastnosti domény ovšem způsobují některé nové problémy.

Pro biomedicínská data je např. typické, že obsahují řadu zkratek a cizích slov z latiny. Složitá je identifikace jednotlivých významových jednotek, protože jsou často složeny z několik slov. Jedná se např. o chemické sloučeniny či názvy nemocí. Word2Vec zpracovává každé slovo z korpusu jako term, mnohoslovné termy nemohou být reprezentovány ve vektorovém prostoru. Při předzpracování je nutné tyto termíny identifikovat a jednotlivá slova spojit např. podtržítkem tak, aby s nimi při procesu učení bylo zacházeno jako s jediným slovem.

6.2 Použité nástroje

Jak již bylo avizováno, nejdůležitějším nástrojem v této práci je Word2Vec. Zatímco o některých jeho alternativách se tvrdilo, že tento nástroj překonávají z hlediska přesnosti modelu k reprezentaci slov vytvořeného učením bez učitele, využil jsem právě Word2Vec, protože z něj vychází i další práce, a proto s nimi budu moc snadněji porovnat dosažené výsledky v další kapitole.

Dále jsem použil tagger Genia. Jak bude demonstrováno v následující kapitole 7, tento nástroje se ukázal jako vhodný prostředek pro dělení souvislého textu na tokeny, což je v biomedicínské doméně jedna z klíčkových částí [1]. Ačkoliv se vyskytly i přesnější tokenizátory, zaujala mě možnost využít jej k rozpoznávání jmenných entit a poté i jako tagger. Jak se ukázalo, v této oblasti dosahuje zajímavých výsledků. Implementace probíhala v jazyce python, vzhledem k možnosti využití řady knihoven pro NLP a implementovanému jazyku samotného Word2Vec.

6.3 Implementace

Nejprve je třeba extrahovat veškerou textovou informaci obsaženou v záznamech dokumentů, v tomto případě se tedy jedná především o abstrakty, případně lze využít také názvy článků a časopisů. Tuto činnost provádí skript `extract_text.py`.

Znaky ze sady Unicode jsou přemapovány na odpovídající náhradu z ASCII. Tento krok je důležitý z hlediska množství běžně používaných NLP nástrojů, které nemají ošetřený vstup v podobě znaků zakódovaných do Unicode, a také jako normalizace získaná z mapování, např. znak β na ASCII řetězec *beta*. Jedná se o časté případy v čistém textu. Získáme tak soubory s plain textem obsahující každý abstrakt či jiný text na samostatném řádku.

Extrahovaný text je poté tokenizován s využitím nástroje Genia, využil jsem i jeho další funkce k rozpoznávání biomedicínských entit a POS taggingu k následnému určení slov k odstranění. Výsledkem je nový soubor se všemi abstrakty, ale již bez interpunkčních symbolů, závorek a jiných nevýznamových znaků. Všechna slova jsou oddělena mezerou. Z původního korpusu jsem tak obdržel přes 2 miliardy tokenů, které jsou tvořeny asi 1 250 000 odlišnými jednotkami o různých frekvencích. Přesný počet závisí na parametrech spuštění. Tabulka 6.2 ukazuje statistiku výskytu jednotlivých elementů v databázi s počtem tokenů v nich.

Element	Počet výskytů	Počet tokenů v elementu
Záznam	23 343 329	2 024 965 229
Název článku	23 343 329	264 705 356
Název časopisu	23 343 329	116 451 859
Abstrakt	11 514 344	1 908 512 045

Tabulka 6.2: Počty jednotlivých elementů v databázi. Počet tokenů byl určen jednoduchou metodou na základě výskytu mezer.

Další skript, `train_model.py`, slouží k ovládání nástroje Word2Vec. U něj jsem využil možnost identifikace víceslovných termínů ze statistiky výskytů n-gramů, čímž dojde k rozšíření slovníku. Poté se získaná data aplikují na učení Word2Vec s vhodně nastavenými parametry spuštění, jejichž vliv na přesnost výsledků budeme dále dokumentovat. Prostřednictvím argumentů příkazového řádků umožňuje trénovací skript nastavení těchto parametrů učícího procesu Word2Vec:

- typ architektury – CBOW nebo SG
- dimenzionalitu vektorového prostoru
- velikost kontextového okénka v počtech slov
- trénovací algoritmus – hierarchical softmax nebo negative sampling
- práh převzorkování frekventovaných slov
- počet použitých vláken
- minimální počet výskytů slova
- formát výstupu

Výstupem skriptu je vektorová reprezentace, kterou lze načíst a testovat např. prostřednictvím dodaného skriptu `toolkit.py`. Součástí odevzdávaného nosiče je také soubor s návodem.

6.4 Požadavky na spuštění

Pro spuštění skriptu `toolkit.py`, určeného k analýze vytvořeného modelu jsou vyžadovány následující prostředky:

- interpret pro jazyk python verze 2.6 nebo vyšší
- Word2Vec

K vytvoření dalších modelů tímto způsobem je také vyžadován Genia tagger.

Kapitola 7

Porovnání tokenizátorů

Jak už bylo řečeno, tokenizace biomedicínských dat klade vyšší nároky než je tomu u obecných textů, jelikož se v nich vyskytují netypické jazykové konstrukce, jako jsou genové sekvence, chemické sloučeniny, názvy proteinů a další. Chyby v tomto kroku mohou negativně ovlivnit výsledky dalších etap, v tomto případě se jedná zejména o strojové učení. Některé tokenizátory např. rozdělují slova v místě výskytu čísla na dva a více tokenů, ale jiné je ponechávají dohromady jako jeden celek.

Uvážíme-li, že u biomedicínských dat může být vstupem tokenizátoru např. `H2SO4`, prvním přístupem by došlo k roztržení významové jednotky na `H`, `2`, `SO`, `4`. Takové tokeny mohou být během procesu učení nesprávně spojovány s dalšími entitami. V této kapitole se proto zaměříme na analýzu některých tokenizátorů při zpracovávání biomedicínského textu. Existuje několik studií věnujících se srovnávání tokenizátorů pro obecné texty, ovšem pro takto specifickou doménu jsou analýzy vzácné.

Výsledky ukazují, že výstupy jednotlivých tokenizátorů se výrazně liší a volba toho správného pro konkrétní úlohu vyžaduje podrobné informace. Analýza je proto tvořena i za účelem pomoci rozhodovacímu procesu případného čtenáře. Připomeňme jenom, že v následujícím textu termín *slovo* odkazuje k lexikografickým jednotkám v původním textu, zatímco *token* značí menší kusy textu tvořící slova.

7.1 Porovnávané nástroje

Celkem jsem porovnal 9 tokenizačních prostředků, z nichž řada je dostupná jako open source software. Všechny testované nástroje, včetně dále používaných čísel k jejich identifikaci, jsou uvedeny v tabulce 7.1. Některé z nich jsou přednostně POS taggery nebo automaticky provádí i řadu dalších operací, ale mohou být využity k tokenizaci jako k jedné z funkcí. Z těchto důvodů nemá příliš smysl měřit jejich rychlost. Za účelem porovnání tokenizerů navzájem jsem odstranil tagy a jiné nevyžadované výstupy, pokud byly produkovány automaticky.

Když nástroj nabízel dva nebo více tokenizátorů, vybral jsem pouze ten, který produkuje lepší výsledky nebo nevyžaduje další úsilí ke spuštění. Výjimkou z tohoto pravidla je NLTK tokenizer, u kterého jsem na základě častého využití otestoval 2 dostupné tokenizátory (TreebankWordTokenizer, WhitespaceTokenizer). V případě tokenizerů, které vyžadují trénovací sadu, jsem použil jazykový prostředek poskytovaný s daným nástrojem.

Číslo	Tokenizátor	Verze	Jazyk	Trénovací mód
1	NLTK TreebankWordTokenizer [4]	2.0.4	Python	Ne
2	NLTK WhitespaceTokenizer [4]	2.0.4	Python	Ne
3	Stanford POS Tagger [24]	3.4.1	Java	Ano
4	Mallet tokenizer [16]	2.0.7	Java	Ne
5	OpenNLP tokenizer [3]	1.5.3	Java	Ano
6	Lingpipe [2]	4.1.0	Java	Ano
7	Genia tagger [30]	3.0	C++	Ne
8	MetaMap [26]	2014	Java	Ne
9	MedPost/SKR POS Tagger [25]	3.0	Java	Ano

Tabulka 7.1: Porovnávané nástroje.

7.2 Testovací sada

Nyní vytvoříme testovací sadu, kterou použijeme k měření výstupů tokenizátorů. Pokud chceme podat komplexní analýzu, musíme se nejprve zaměřit na problémy obecného textu, např. na slova obsahující jeden nebo více typů interpunkce. Použijeme je k demonstraci odlišností mezi tokenizátory z pohledu interpunkčního dělení. Poté bude věnována pozornost specifičtějším případům, jako je určování hranic u termínů chemických sloučenin. Tyto případy demonstrují rozdílnosti tokenizátorů z pohledu rozpoznávání jmenných entit. Celá sada je uveden v příloze A.

Čtenáři hledající vhodný tokenizátor získají přehled, čemu u každého z nich věnovat zvláštní pozornost. Pokud navíc plánují využít některý se zde uvedených, jednoduše mohou vyloučit ty, které nevyhovují jejich konkrétním požadavkům. Vzhledem k omezenému rozsahu, který můžu věnovat této problematice, nelze pokrýt komplexnější případy jmenných entit, jako jsou názvy organizaci, emailové adresy a řady dalších problémů. V těchto případech může být vhodné, pokud tokenizátor dokáže nejprve rozpoznat jmenné entity a použít informaci k provedení tokenizace.

7.3 Porovnání výstupů

Jak jsme již poznamenali, tokenizátory činí různá rozhodnutí v tom, co tvoří token. V této sekci se proto více zaměříme na odlišná tokenizační schemata. Podíváme se na jejich výstupy a porovnáme je mezi sebou. Prodiskutujeme zejména vlastnosti důležité pro tuto práci, avšak kompletní výstupy pro testovací sadu jsou uvedeny v příloze A. Z výsledků vyplývají dvě hlavní tokenizační schemata:

- dělení na tokeny pouze dle výskytu mezer
- dělení na tokeny pouze dle výskytu mezer, čísel a interpunkce

Toto rozdělení neplatí úplně, najde se několik výjimek. Např. pro vstup `isn't` zřejmě existuje nějaké pevně zadané pravidlo navíc, které slovo rozdělí na `is` a `n't`, kde `n't` je bráno jako speciální forma `not`. Při rozdělování textu na tokeny nejsou místa dělení vždy stejná dokonce ani pro slova ze stejné kategorie testování. Většina tokenizátorů nesleduje jenom toto základní pravidlo a jejich výstupy se liší, když text obsahuje slova spojená pomlčkou, desetinná čísla, zkratky a další.

7.3.1 Čísla s interpunkcí

Čísla s interpunkcí se v textu vyskytují jako desetinná čísla (`-67.8`), časové údaje (`11:20`), intervaly (`17-19`), zlomky (`3/4`), velká čísla (`51,000`) atd. Tyto prvky můžeme různě kombinovat a získáme tak nejrůznější výrazy. Většina tokenizátorů uchovává čísla jako celek nebo je rozděluje na místě výskytu interpunkce. Výjimkou je tokenizátor 4, který na výstupu nemá žádné tokeny.

U výrazů existuje obvykle několik způsobů, jak je tokenizovat. Jelikož operátor značí zvláštní význam a prioritu, správnost jednotlivého tokenazičního přístupu je méně subjektivní v porovnání s běžným textem. Např. výstupy tokenizátorů 6, 7 pro vstup `11:20-11:30%` jsou nežádoucí. Symbol ␣ značí hranice sousedních tokenů.

- `-67.8`
  - `-67.8` (beze změny): tokenizátory 1, 2, 3, 7
  - `-67`␣`.`␣`8`: tokenizátor 6
  - `-`␣`67`␣`.`␣`8`: tokenizátory 5, 8, 9
  - žádný výstup: tokenizátor 4
- `17-19`
  - `17-19` (bez změny): tokenizátory 1, 2, 3, 6, 7
  - `17`␣`-`␣`19`: tokenizátory 5, 8, 9
  - žádný výstup: tokenizátor 4
- `99%`
  - `99%`: tokenizátory 1, 2, 3, 6
  - `99`␣`%`: tokenizátory 5, 7, 8, 9
  - žádný výstup: tokenizátor 4
- `11:20-11:30`
  - `11:20-11:30` (beze změny): tokenizátory 1, 2
  - `11:20`␣`-11:30`: tokenizátor 3
  - `11`␣`:`␣`20-11`␣`:`␣`30`: tonenizátor 6, 7
  - `11`␣`:`␣`20`␣`-`␣`11`␣`:`␣`30`: tonenizátory 5, 8, 9
  - žádný výstup: tokenizátor 4
- `44.9+/-3.0`
  - `44.9+/-3.0` (bez změny): tokenizátory 1, 2, 7
  - `44.9`␣`+`␣`/`␣`-3.0`: tokenizátor 3
  - `44`␣`.`␣`9`␣`+`␣`/`␣`-3`␣`.`␣`0`: tokenizátor 6
  - `44`␣`.`␣`9`␣`+`␣`/`␣`-`␣`3`␣`.`␣`0`: tokenizátory 5, 8, 9
  - žádný výstup: tokenizátor 4

7.3.2 Slova s čísly

Jedná se o slova běžné angličtiny i chemické vzorce. Tokenizátory 4 a 5 nenechávají čísla a písmena dohromady – rozdělují `H2O` na více částí, tokenizátor 5 jako jediný dělí `1990s` na `1990` a `s`. Na druhou stranu číslo 4 odstraňuje číselnou část z anglicky psaných řadových číslovek. Všechny ostatní tokenizátory ponechávají tato slova jako jednu jednotku.

- `20th`
  - `20th` (bez změny): tokenizátory 1, 2, 3, 6, 7, 8, 9
  - `20`␣`th`: tokenizátor 5
  - `th`: tokenizátor 4
- `cm2`
  - `cm2` (bez změny): tokenizátory 1, 2, 3, 6, 7, 8, 9
  - `cm`␣`2`: tokenizátor 5
  - `cm`: tokenizátor 4

7.3.3 Slova s interpunkcí

Základní funkce interpunkce v písemném projevu spočívá ve vyjádření struktury a organizace textu, slovo s tečkou na konci tedy většinou značí konec věty. Avšak tečka se dále objevuje ve zkratkách, jako jsou (`etc.`, `I.Q.`). Pro zkratky existují 4 varianty přístupu. Tokenizátory 2, 3 je ponechává beze změny, 1 a 7 odebírají pouze poslední tečku podobně jako u konce věty. 5, 6, 8 a 9 tečky dělí na samostatné tokeny a 4 v místě tečky dělí slovo.

- `etc.`
  - `etc.` (bez změny): tokenizátory 2, 3
  - `etc`␣`.`: tokenizátory 1, 5, 6, 7, 8, 9
  - `etc`: tokenizátor 4
- `p.a.`
  - `p.a.` (bez změny): tokenizátory 2, 3
  - `p.a`␣`.`: tokenizátory 1, 7
  - `p`␣`a`: tokenizátor 4
  - `p`␣`.`␣`a`␣`.`: tokenizátory 5, 6, 8, 9

Apostrofy mohou značit vlastnictví (`woman's`), spojování slov zkrácením (`isn't`), jména (`O'Doole`), citace (`'right way'`) atd. Existuje pro ně řada tokenizačních přístupů:

- `Texas'`
  - `Texas'` (bez změny): tokenizátory 2, 6
  - `Texas`␣`'`: tokenizátory 1, 3, 5, 7, 8, 9
  - `Texas`: tokenizátor 4

- `isn't`
  - `isn't` (bez změny): tokenizátory 2, 6
  - `is␣n't`: tokenizátory 1, 3, 7
  - `isn␣'␣t`: tokenizátory 5, 8, 9
  - `isn␣t`: tokenizátor 4
- `O'Doole`
  - `O'Doole` (bez změny): tokenizátory 1, 2, 3, 6, 7
  - `O␣'␣Doole`: tokenizátory 5, 8, 9
  - `O␣Doole`: tokenizátor 4

U slov spojených pomlčkou (spojovníkem) se opakují dvě hlavní schemata – ponechat celé spojení jako jeden celek nebo rozdělit slova na dva či více tokenů. Tokenizátory 1, 2, 3, 6, 7 používají první strategii, 5, 8, 9 druhou. Stejně je tomu i u lomítek, jenom tokenizátor 6 zde využívá přístup dělení. Přístup s nedělením slov obsahujících lomítky selhává, pokud je komponenta na kterékoliv straně od lomítka tvořena slovy spojenými pomlčkou. Zbylý nástroj, to je číslo 4, je úplně odlišný, jelikož odstraňuje pomlčky i lomítka z výstupu:

- `Hi-Tech`
  - `Hi-Tech` (bez změny): tokenizátory 1, 2, 3, 6, 7
  - `Hi␣-␣Tech`: tokenizátory 5, 8, 9
  - `Hi␣Tech`: tokenizátor 4
- male/female
  - `male/female` (bez změny): tokenizátory 1, 2, 3, 7
  - `male␣/␣female`: tokenizátory 5, 6, 8, 9
  - `male␣female`: tokenizátor 4

Bez ohledu na typ závorek, téměř všechny tokenizátory oddělují kulaté a hranaté závorky od slov uvnitř závorek, jenom číslo 3 konvertuje závorky na jiné symboly. Tokenizátor 4 odstraňuje závorky úplně. V případě nástrojů 1, 5, 6, 7, 8, 9 jsou závorky vždy odděleny od slova, 2 jako jediný ponechává vstup beze změny.

- `(atom)`
  - `(atom)`: tokenizátor 2
  - `(␣atom␣)`: tokenizátory 1, 2, 5, 6, 7, 8, 9
  - `-LRB-␣atom␣-RRB-`: tokenizátor 3
  - `atom`: tokenizátor 4

7.3.4 Biomedicínské termíny

Následující příklady ilustrují jemnější rozdíly mezi tokenizátory z pohledu rozpoznávání jmenných entit. V těchto případech zahrnuje každý vstup víc než jeden typ interpunkce a kde místo rozdělení závisí na prioritě zpracovávání interpunkce a pořadí v tokenizátoru.

- `H2SO4`
  - `H2SO4` (bez změny): tokenizátory 1, 2, 3, 6, 7, 8, 9
  - `H␣ 2␣SO ␣4`: tokenizátor 5
  - `H␣ SO`: tokenizátor 4
- `5,6-dihydrodiol-3-methyl-2-oxo-1,2-dihydroquinoline`
  - `5,6-dihydrodiol-3-methyl-2-oxo-1,2-dihydroquinoline` (bez změny): tokenizátory 1, 2
  - `5,6-dihydrodiol-3-methyl-2-oxo-1␣,2␣-␣dihydroquinoline`: tokenizátor 3
  - `5␣,␣6-dihydrodiol-3-methyl-2-oxo-1␣,␣2-dihydroquinoline`: tokenizátory 6, 7
  - `5␣,␣6␣-␣dihydrodiol␣-␣3␣methyl␣-␣2␣-␣oxo␣-␣1␣,␣2␣-␣dihydroquinoline`: tokenizátor: 8
  - `5␣,␣6␣-␣dihydrodiol␣-␣3␣-␣methyl␣-␣2␣-␣oxo␣-␣1␣,␣2␣-␣dihydroquinoline`: tokenizátory 5, 9
  - `dihydrodiol␣methyl␣oxo␣dihydroquinoline`: tokenizátor 4
- `5'-AAAGTCT-3'`
  - `5'-AAAGTCT-3'` (bez změny): tokenizátory 2, 5
  - `5'-AAAGTCT-3␣ '`: tokenizátory 1, 7
  - `5␣'␣-␣AAAGTCT-3␣'`: tokenizátor 3
  - `5␣'␣-␣AAAGTCT␣-␣3␣'`: tokenizátory 5, 8, 9
  - `AAAGTCT`: tokenizátor 4

7.4 Vyhodnocení výsledků

V této sekci prodiskutujeme výhody a nevýhody jednotlivých tokenizátorů, které jsme otestovali. Zběžnou analýzou výsledků si můžeme všimnout, že se jedná o pestrou množinu, v které se neobjevují totožné výstupy. Strategie nástrojů s nejmenším počtem tokenů byly podobné, spočívaly v rozdělování pouze podle mezer. Nástroje s velkým počtem změn naopak využívaly výskyty mezer, čísel i interpunkce. Mallet tokenizátor také odstranil všechnu interpunkci a čísla z výstupu. Výstupy ostatních systémů se lišily, tokenizace zahrnovala interpunkci, složená slova, čísla, závorky a další nealfanumerické symboly, zkratky a různé entity.

Vyhodnocením uvedených případů jsme získali základní porozumění chování každého tokenizátoru. Nyní si stručně popíšeme každý nástroj a vyhodnotíme jeho výsledky.

7.4.1 NLTK tokenizer

NLTK (Natural Language Toolkit) je nástroj pro práci s jazykovými daty napsaný v jazyce python. Jeho cílem je usnadnit výzkum v NLP a úzce souvisejících oblastech, jako je umělá inteligence, lingvistika a strojové učení. Díky velké oblíbenosti je doprovázen řadou návodů, které popisují základní použitelné koncepty zpracování jazyka a způsob jejich využití. NLTK byl vyvinut na Pensylvánské univerzitě. Poskytuje snadno použitelné rozhraní pro klasifikaci, stemming, tagging a další operace, využívá řadu lexikálních prostředků včetně WordNet.

U tohoto nástroje jsem z demonstračních důvodů rozhodl použít dva tokenizátory, konkrétně to jsou TreebankWordTokenizer a WhitespaceTokenizer. Jak je patrné z názvu, druhý jmenovaný dělí text na tokeny pouze dle výskytu mezer, což jej činí nepoužitelným v reálných problémech.

TreebankWordTokenizer je již zajímavější. Ve většině případů naopak nechává vstup beze změny, výjimkou je interpunkce na konci slov. Jedná se o příliš zjednodušený tokenizátor pro oblast dat, na kterou jej chceme aplikovat. Avšak pro uživatele používající jazyk python, ve kterém je napsán, může být základem pro výstavbu komplexnějších systémů.

7.4.2 Stanford POS Tagger

Jak už název napovídá, tento nástroj byl primárně vytvořen jako POS tagger a je vyvíjen na Stanfordské univerzitě, konkrétně v jazyce Java. Software kromě rozhraní prostřednictvím příkazového řádku poskytuje GUI demo, API. Metoda je založena na maximální entropii. K získání výstupů testovací sady jsme spustili tagger s defaultním volně dostupným modelem, ovšem k dispozici jsou také modely pro arabštinu, čínštinu, francouzštinu, španělštinu a němčinu.

Ačkoliv je navrhnut jako POS tagger, lze jej úspěšně využívat i jako tokenizátor obecného textu. Selhává u komplikovanějších výrazů, kombinujících více typů interpunkce, a zejména u slov specifických pro biomedicínskou doménu. Zajímavostí tohoto nástroje je, že převádí závorky na speciální tokeny `-LRB-`, `-RRB-`.

7.4.3 Mallet tokenizer

MALLET (MAchine Learning for LanguagE Toolkit) je integrovaná sbírka kódů v Javě pro statistické NLP, klasifikaci dokumentů, shlukovou analýzu a další aplikace s textem. Kromě aplikací strojového učení MALLET zahrnuje také procedury pro transformaci textových dokumentů do číselné reprezentace, která může být efektivněji zpracována. Tento proces je implementován prostřednictvím flexibilního systému rour, které kontrolují jednotlivé úkoly zpracování. Tokenizace je implementována v balíku CharSequence2TokenSequence.

Vzhledem k výstupům jednotlivých testovacích sad se tento nástroj nejeví jako vhodný pro jakýkoliv úkol v oblasti NLP. Odebírá ze vstupu jakýkoliv nepísmenný symbol včetně čísel, která ovšem mohou mít důležitý význam. Pokud se ve slově objeví nějaký takový znak, slovo je v tomto místě pravidelně rozděleno. Tato metoda generuje jenom velmi jednoduché výsledky, které nemohou najít uplatnění snad v žádné aplikaci.

7.4.4 OpenNLP tokenizer

OpenNLP je open source projekt obsahující široké množství NLP nástrojů, zahrnující algoritmy strojového učení založené na maximální entropii a perceptronu. Má velmi dobré

programové rozhraní, takže jej lze velmi snadno integrovat do programů v Javě, díky čemuž je poměrně populární. Zajímavostí je, že umožňuje trénování modelů a poskytuje Java API. Tento nástroj jsem testoval s defaultním modelem.

Tokenizátor obsahuje gramatický model, který lze trénovat. Uživatel může jak trénovat systém s vlastní trénovací sadou, tak aplikovat syntaktický model poskytovaný s OpenNLP systémem. Patří mezi tokenizátory s největším počtem tokenů, což znamená, že slova jsou často dělena i tam, kde si to uživatel nepřeje. Nevýhodu můžeme spravit použitím odlišného trénovacího jazykového modelu.

7.4.5 Lingpipe

Jedná se o soubor Java knihoven pro NLP, podporuje hned 3 typy chunkerů - založené na pravidlech, slovníku a statistice. Pro tokenizaci jsem využil ten statistický, založený na skrytých Markovových modelech, protože dle dokumentace se jedná o nejpřesnější typ. Efektivní, robustní architektura Lingpipe umožňuje snadnou integraci s jinými systémy. Podobně jako u OpenNLP i zde je možné trénování, existuje hned několik předkompilovaných modelů pro různé jazyky.

Lingpipe patřil k nástrojům s nejpestřejšími výstupy. Zatímco jednoduché čísla s jedním typem interpunkce jsou prakticky vždy rozdělena na více tokenů, slova s pomlčkami či apostrofy jsou ponechána v původním tvaru. To má za následek, že např. `5'-AAAGTCT-3'` zůstává v jediném tokenu, ačkoliv řada jednodušších entit, u kterých bychom o to více čekali stejné zacházení, je pozměněna.

7.4.6 Genia tagger

Tento nástroj analyzuje anglické věty a jeho výstupem jsou základní tvary slov, POS tagy, tagy jmenných entit atd. Je však určen speciálně pro biomedicínské texty jako jsou právě MEDLINE abstrakty, což by v tomto testování mělo být značnou výhodou. K vytvořeným tokenům je schopen přiřadit taktéž informaci o typu entity – protein, DNA, RNA atd. Genia tagger využívá algoritmus založený na modelu maximální entropie.

Tento nástroj spíše ponechával slova v jednom tokenu, než že by je přehnaně dělil, čemuž odpovídá i nízký počet celkových tokenů. Slova s pomlčkou, lomítkem, apostrofem nebo čísly ponechává dohromady, problémy způsobují zejména tečky a čárky. Ačkoliv jeho výstup nepovažuji za nejlepší, rozhodl jsem se v další práci využít právě tento nástroj, protože provádí také určitou formu rozpoznávání biomedicínských entit.

7.4.7 MetaMap

MetaMap je vysoce konfigurovatelný program vyvíjený jako součást NLM. Jedná se o sbírku Java objektů pro analýzu textových dokumentů a identifikování slov, termů, frází atd. MetaMap umožňuje nastavení řady parametrů, které ovlivňují množství informací na výstupu. Je to jeden ze základů medicínského indexování textu, které se používá k poloautomatickému a plně automatickému indexování biomedicínské literatury na NLM. Zajímavostí je, že jej lze využít i prostřednictvím web API.

Jako mnoho jiných rozděluje i tento tokenizátor text na tokeny podle umístění mezer a interpunkce, počet takto vytvořených tokenů patří mezi ty největší. Jeho výhodou však je, že umožňuje zpětně sestavit původní dotaz, a to použitím další modulu tohoto nástroje. Další typickým rysem tokenizátoru je, že na výstupu nalezneme také informaci o počtu mezer mezi sousedními tokeny, což znovu napomáhá rekonstrukci tokenů na originální text.

7.4.8 MedPost/SKR POS Tagger

Tento nástroj v implementačním jazyce Java je určen pro biomedicínské texty, jako trénovací korpus byly použity právě abstrakty MEDLINE. Tagger byl původně vyvíjen skupinami z National Center for Biotechnology Information a Lister Hill National Center for Biomedical Communications v kombinaci jazyků C++ a perl. Využívá skryté Markovovy modely, kombinuje kontextovou a lexikální informace ke zlepšení přesnosti značení. I když je určen k POS taggingu, ale jeho volba `-token` s ním umožňuje zacházet jako s tokenizátorem.

I přes své určení pro biomedicínská data zbytečně dělí i tyto entity. Jeho přístup spočívá v dělení jakéhokoliv slova dle výskytu všech jiných znaků, než jsou písmena a čísla. Jeho výhodou je, že je schopen zpracovat Unicode znaky.

Kapitola 8

Testování výsledných modelů

V předchozí kapitole jsme vytvořili reprezentaci vektorového prostoru nyní se blíže podíváme na její ohodnocení. V první části porovnáme nejen námi vytvořené modely, ale také je poměříme s existujícími. Očekáváme, že v oblasti biomedicíny by měl dosahovat lepších výsledků, než obecné modely. V druhé části testování na vybraném modelu ověříme, jestli splňuje požadavek zadání na hledání synonym.

Výsledky značně ovlivňují použitelnost. Má-li být model prospěšný širší společnosti, je třeba, aby spolehlivě určoval sémantické vztahy, pro jejichž zachycení byl vytvořen. Chyby výstupů může způsobovat několik faktorů. Tím prvním jsou nedostatky samotného korpusu. Můžou v něm převažovat data spadající k určitému tématu, čímž se zhorší schopnost zachytit ostatní vztahy nebo může korpus obsahovat nedostatečné množství dat. Vzhledem k velikostem běžných korpusů se jen obtížně detekují původci problémů.

Často se používá analýza n-gramů, jejíž cílem je zjistit počty spoluvýskytů určitých slov a odhalit tak, zda se daný vztah v korpusu vůbec vyskytuje. Také nelze vyloučit, že i v samotném korpusu se nachází množství překlepů a chyb, ale vzhledem k velkému množství záznamů je možné tyto chyby zanedbat. Další chyby může způsobovat samotný učící nástroj, jelikož vytváření vektorů z velkého množství dat není jednoduché.

8.1 Porovnání s existujícími modely

V této sekci porovnáme námi vytvořené modely s dalšími dostupnými předtrénovanými vektorovými reprezentacemi. První z nich bude model Google zmiňovaný již v části 5.4.4. Věřím, že na základě porovnávání s modelem pro obecná data vyniknou schopnosti reprezentací určených pro specifickou doménu.

Dále se mi podařilo najít dva modely přímo pro biomedicínskou oblast, tudíž získám i porovnání se stejně zaměřenými nástroji. Tyto modely jsem získal z Biomedical natural language processing[1] (BNLP), což je web poskytující různé nástroje pro zpracování biomedicínských dat. Vektory slov byly vytvořeny s využitím nástroje nxml2text k předpracování a Word2Vec s dimenzí 200. Modely se liší velikostí použitých trénovacích dat – první model byl trénován na 2 896 348 481 tokenech, druhý na 5 487 486 225. Z tohoto pohledu mají konkurenční reprezentace výhodu.

Vzhledem k počtu vytvořených modelů zde nelze uvést všechny výsledky, proto jsem vybral 3 zástupce. Budu je dále označovat jako 200_5, 200_10 a 300_10, kde číslo před

[1] Dostupné na adrese https://code.google.com/p/word2vec/.

podtržítkem značí použitou dimenzi vektoru a číslo za ním velikost kontextového okénka. Na analýzu vlivu těchto parametrů jsem se soustředil nejvíce.

K porovnání těchto modelů jsem vytvořil tři jednoduché testovací sady pro funkce Word2Vec, obsahující vždy 25 příkladů. Testy se skládají z nejrůznějších pojmů spadajících do oblasti biomedicíny, které jsem si vhodně zvolil tak, aby umožňovaly otestování všech modelů. Model od Google byl trénován na obecných datech, tudíž jeho slovník neobsahuje některé specifické termíny (názvy proteinů, latinské výrazy, doménové zkratky atd.). Z tohoto důvodu jsem v této části negeneroval náhodně vybrané termíny.

8.1.1 Test 1

První testovací sada je postavena na úloze, kdy pro danou množinu slov máme určit, které její slovo významově nepatří mezi ostatní. Např. mezi slova *adenine, thymine, cytosine, retinol* nepatří *retinol*, protože se jedná o vitamín, zatímco ostatní slova značí nukleotidy. Tato úloha je ve Word2Vec implementována prostřednictvím funkce `doesnt_match`. Výsledky této části ukazuje tabulka 8.1.2, chybně vyřazená slova jsou označena podtržením.

Můžeme si všimnout, že nejlepších výsledků dosahují modely 300_10 (100 %) a BNLP2 (96 %). Ukazuje se, že velikost trénovacího korpusu a dimenze vektorového prostoru hrají velkou roli. Nejméně úspěšný byl model od Google (jen 52 %), jenž měl problémy i s jednoduššími případy. Dle výsledků je tedy Word2Vec schopen zachytit všechny sémantické souvislosti, nicméně pro vektory menších dimenzí existují chybné výsledky.

8.1.2 Test 2

Dalším testem bude odhalování analogií. Na vstupu máme tři slova, budeme je značit A, B, C, taková, že mezi A a B je nějaký vztah. Úkolem je doplnit čtvrté slovo D tak, aby mezi A, B a C, D byly analogické vztahy. Např. pro slova *testosterone, man, estrogen* vydedukujeme chybějící slovo *woman*, protože testosteron je pohlavní hormon mužů, zatímco estrogen žen. Tuto úlohu symbolizuje funkce `most_similar` z Word2Vec, nechal jsem si pokaždé vypsat 10 nejlepších odhadů pro vstupní trojici.

Ačkoliv jsem se tomu při tvorbě této sady snažil vyhnout, v některých případech lze doplnit více než jedno slov tak, aby analogie dvojic zůstala přesně zachována (*fat* a *overweight*, *alcohol* a *ethanol*). Podobně, jako drobné syntaktické rozdíly (*thymine* a *thymine_T*, *woman* a *women*), jsem se rozhodl tyto výstupy uznat jako správnou odpověď. To nicméně znamená, že tuto část nelze vyhodnocovat automaticky.

V tabulce 8.1.2 jsou uvedeny výsledky této části. Číslo znamená, kolikátý nejlepší odhad daného modelu byl požadovaný výstup, symbol • značí, že mezi deseti nejlepšími výstupy nebyl nalezen takový, který by mohl být považovaný za správný. Z tabulky vyplývá, že všechny modely dokázaly několikrát vrátit požadovaný termín hned na prvním místě. To znamená, že Word2Vec je schopen poměrně přesně zachytit analogické vztahy. Námi vytvořené modely v této části poměrně jasně zvítězily, zajímavé ovšem je, že lépe dopadly ty s omezenějším rozměrem vektorů.

Nečekaně dobře si vede i obecný model, který překonal i obě reprezentace BNLP, jejichž výstup z této části je velké zklamání, o čemž vypovídá velký počet nenalezených analogií. Pro 10 z 25 trojic se oběma vůbec nepodařilo najít požadovaný výstup. Z případů, které dělaly největší problémy, můžeme zmínit *milk, lactose, fruit, fructose*, protože jej žádný z modelů nevyhodnotil ani částečně správně. Samozřejmě nelze vyloučit ani možnost nevhodného zadání.

Vstup	**Google**	**BNLP1**	**BNLP2**	**200_5**	**200_10**	**300_10**
retina pupil cornea cerebrum	• pupil	cerebrum	cerebrum	cerebrum	cerebrum	cerebrum
nucleus ribosome vacuole chlorophyll	• nucleus	chlorophyll	chlorophyll	chlorophyll	chlorophyll	chlorophyll
obesity anorexia bulimia stroke	stroke	stroke	stroke	stroke	stroke	stroke
protease lipase nuclease disease	disease	disease	disease	disease	disease	disease
adenine thymine cytosine retinol	retinol	retinol	retinol	retinol	retinol	retinol
LSD heroin methamphetamine penicillin	penicillin	penicillin	penicillin	penicillin	penicillin	penicillin
testosterone estrogen progesterone prolactin	• testosterone	• estrogen	• estrogen	prolactin	prolactin	prolactin
cancer tumor cigarette sugar	sugar	sugar	sugar	sugar	sugar	sugar
sugar diabetes insulin tobacco	tobacco	tobacco	tobacco	tobacco	tobacco	tobacco
myocardium coronary heart intestine	• heart	intestine	intestine	intestine	intestine	intestine
aspirin penicillin antibiotic fever	fever	fever	fever	fever	fever	fever
depression sadness anxiety tobacco	tobacco	tobacco	tobacco	tobacco	tobacco	tobacco
chitin insect biopolymer uracil	• insect	• insect	uracil	• insect	• insect	uracil
cholesterol lipid fat diabetes	• fat	diabetes	diabetes	diabetes	diabetes	diabetes
tibia fibula femur lisp	lisp	lisp	lisp	lisp	lisp	lisp
exon transcript gene infection	• transcript	infection	infection	infection	infection	infection
fermentation glucose ATP acid	• ATP	acid	acid	• ATP	• ATP	acid
hydrogen deuterium tritium calcium	calcium	calcium	calcium	calcium	calcium	calcium
dopamine adrenaline thyroxine estrogen	• adrenaline	• thyroxine	estrogen	• thyroxine	estrogen	thyroxine
nanotechnology synthesis engineering health	health	health	health	health	health	health
cadmium mercury chrome oxygen	• chrome	oxygen	oxygen	oxygen	oxygen	oxygen
keratin hair nail pancreas	• nail	pancreas	pancreas	pancreas	pancreas	pancreas
atlas axis vertebra muscle	• atlas	muscle	muscle	muscle	muscle	muscle
embryo fetus germ adolescent	adolescent	adolescent	adolescent	adolescent	adolescent	adolescent
fructose glucose monosaccharide caffeine	caffeine	caffeine	caffeine	caffeine	caffeine	caffeine
Úspěšnost	**52 %**	**88 %**	**96 %**	**88 %**	**92 %**	**100 %**

Tabulka 8.1: Výsledky prvního testu, nepatřící slovo je vždy uvedeno jako poslední.

A	B	C	D	Google	BNLP1	BNLP2	200_5	200_10	300_10
RNA	uracil	DNA	thymine	5	1	1	1	1	1
anorexia	slim	obesity	fat	•	1	2	2	2	1
plus	cation	minus	anion	1	•	•	1	3	1
testosterone	man	estrogen	woman	1	1	1	1	1	1
thiamine	riboflavin	B1	B2	1	3	1	1	1	1
vertebra	spine	cerebrum	brain	1	2	2	4	2	4
artery	vein	output	input	1	2	2	1	1	1
milk	lactose	fruit	fructose	•	•	•	•	•	•
cigarette	nicotine	beer	alcohol	•	1	•	1	1	•
erythrocyte	leucocyte	red	white	5	•	•	5	5	•
C	scurvy	D	rickets	•	2	3	1	4	4
promotor	terminator	start	end	4	•	6	•	7	•
spider	arachnophobia	height	acrophobia	1	•	•	•	•	•
finger	arm	toe	leg	4	1	2	1	1	1
headache	depression	tumor	cancer	3	•	•	•	•	•
trypsin	pancreas	pepsin	stomach	•	2	1	1	1	2
stroke	brain	coronary	heart	•	•	•	•	•	•
acid	nonmetal	hydroxide	metal	•	5	8	1	9	2
virus	disease	HIV	AIDS	1	•	•	•	•	1
erythrocyte	anemia	leukocyte	thrombocytopenia	•	3	3	4	3	•
uracil	adenine	cytosine	guanine	2	•	•	6	10	5
female	ovum	male	sperm	7	•	•	1	4	4
pellagra	B3	beriberi	B1	2	7	2	2	1	1
mouth	pharynx	nose	larynx	•	3	6	10	5	10
atrium	heart	cerebrum	brain	1	1	1	1	1	1
Počet bez nálezu				**9**	**10**	**10**	**6**	**5**	**8**

Tabulka 8.2: Výsledky druhého testu. Číslo značí pořadí, na jaké bylo nalezeno požadované slovo. Pokud k nálezu nedošlo, je buňka vyplněna •.

8.1.3 Test 3

Poslední test se bude týkat výpočtu sémantické podobnosti dvou synonym nebo slov výrazně souvisejících. K tomu využijeme funkci `similarity`, počítající kosinovou vzdálenost. Např. pro dvojici *alcohol, ethanol* a BNLP1 model Word2Vec vrátí hodnotu 0,61, což by mělo vyjadřovat, že se jedná do jisté míry o synonyma, ale *alcohol* na rozdíl od *ethanol* také označuje celou skupinu sloučenin. Tato část je velice složitá na vyhodnocení, jelikož nemůžeme jednoduše říct, že vyšší hodnoty znamenají přesnější výsledek.

Průměrné hodnoty podobnosti, uvedené společně s výsledky testování v tabulce 8.2, jsou tedy pouze orientační. Zatímco výstupy modelů BNLP a mého jsou většinou podobné, model Google vykazuje větší rozdíly. Např. pro zmiňovanou dvojici alcohol, ethanol dostáváme hodnotu podobnosti 0,28, což je podstatně méně než u ostatních. Naopak pro slova *disease, sickness* dostáváme nejvyšší hodnotu. Může to být způsobeno tím, že ačkoliv se slova na první pohled tváří jako synonyma, je mezi nimi určitá nuance, která hraje významnou roli v speciálně zaměřeném textu.

8.2 Testování nejlepšího modelu

V této části ověříme funkčnost nejlepšího modelu z předchozí části, kterým byl určen 200_10. Ten sice mírně zaostával v prvním testu, ovšem v ostatních dvou patřil mezi nejlepší. Rozhodl jsem se jej upřednostnit před 300_10 i kvůli menší velikosti. Ukázalo se, že vyšší hodnota kontextového okénka způsobovala mírné zlepšení výsledků, ovšem platí to jen do určité hodnoty. Modely větších dimenzi trpí přehnaně detailní reprezentací.

Bylo náhodně vybráno 12 termínů z korpusu, pro které bude vyhodnoceno 10 nejpodobnějších slov určených daným systémem. Vyhodnocením se rozumí rozhodnutí, zda dané slovo skutečně souvisí se vstupním termínem na základě mých znalostí, výsledku vyhledávání slova s termínem pomocí Google, případně MESH[2] (Medical Subject Headings). Tento slovník opět z NLM jsem využil již při předchozích testech, jelikož kromě popisu dotazovaného slova poskytuje také informaci o významově blízkých slovech. Jediným požadavkem na vybrané termíny byl dostatečný výskyt v korpusu (alespoň 100 případů).

Tabulka 8.4 obsahuje seznam vybraných slov spolu s jejich krátkým popisem. Vzhledem ke specifičnosti oboru a omezenému rozsahu práce bohužel nelze rozebrat význam každého získaného slova. Další tabulky obsahují seznam nejpodobnějších slov, ke kterým jsou ve druhém sloupci doplněny hodnoty podobnosti. Symbol • značí chybně určené slovo.

Úspěšnost modelu je měřena jako počet slov, která jsou v daném kontextu opravdu sémanticky blízká danému termínu. Při ručním vyhodnocení dosáhl systém celkové úspěšnosti 93 %, pouze 8 z počtu 120 získaných slov bylo chybných. Výsledky ukazují, že systém je schopen určovat synonyma a související slova tak, jak je požadováno v zadání.

[2]Dostupné na adrese http://www.nlm.nih.gov/mesh/MBrowser.html.

A	B	Google	BNLP1	BNLP2	200_5	200_10	300_10
spine	vertebra	0,53	0,78	0,77	0,67	0,68	0,65
alcohol	ethanol	0,28	0,61	0,49	0,59	0,58	0,55
chloroplast	chlorophyll	0,50	0,50	0,44	0,45	0,53	0,49
overweight	obesity	0,55	0,73	0,71	0,65	0,7	0,70
virus	infection	0,67	0,55	0,56	0,49	0,54	0,48
MDMA	ecstasy	0,64	0,78	0,70	0,78	0,78	0,79
nucleus	core	0,31	0,20	0,28	0,26	0,24	0,20
appendix	intestine	0,36	0,49	0,46	0,44	0,45	0,37
pain	ache	0,63	0,68	0,72	0,45	0,45	0,41
lipid	fat	0,39	0,38	0,38	0,40	0,39	0,36
carcinoma	tumor	0,63	0,48	0,50	0,53	0,59	0,53
gluten	cereal	0,42	0,59	0,55	0,49	0,55	0,51
habitat	territory	0,25	0,30	0,29	0,35	0,35	0,34
disease	sickness	0,43	0,17	0,22	0,19	0,22	0,16
skull	cranium	0,65	0,81	0,84	0,83	0,82	0,81
carbohydrate	saccharide	0,48	0,66	0,65	0,69	0,69	0,63
gamete	ovum	0,56	0,71	0,69	0,56	0,59	0,53
marijuana	cannabis	0,81	0,93	0,94	0,92	0,91	0,88
excavation	paleontology	0,41	0,22	0,25	0,30	0,38	0,33
glycerol	glycerin	0,62	0,57	0,66	0,60	0,55	0,48
embryo	zygote	0,63	0,73	0,76	0,72	0,75	0,70
HIV	AIDS	0,76	0,75	0,71	0,62	0,70	0,68
osmosis	diffusion	0,28	0,32	0,32	0,25	0,29	0,23
heroin	diacetylmorphine	0,48	0,62	0,68	0,59	0,65	0,62
obesity	fatness	0,61	0,59	0,56	0,53	0,56	0,51
Průměr		**0,52**	**0,57**	**0,57**	**0,53**	**0,56**	**0,51**

Tabulka 8.3: Výsledky třetího testu.

Slovo	Krátký popis
KCNQ1	lidský protein
humuli	taxonomické označení druhu organismů
straining	můžeme přeložit jako napětí, natažení, přetížení
Oryzias	rod ryb žijících převážně v Asii
Lactate	součást kyseliny mléčné, proces laktace
paravertebral	paravertebrální, umístěný podél obratlů
APV	2-amino-5-phosphonopentanoic, NMDA receptor
BTG2	lidský protein patřící do rodiny BTG
dimple	můžeme přeložit jako důlek
satiety	přesycení, nasycení
LDPE	Low-density polyethylene, thermoplast
premaxillary	umístění před horní čelistí

Tabulka 8.4: Seznam testovaných slov s jejich krátkým popisem.

Slovo	Podobnost
KCNE1	0,91
KCNE2	0,89
KCNE3	0,87
KvLQT1	0,82
KCNH2	0,82
Kv7.1	0,80
KCNE	0,78
MiRP1	0,78
SCN5A	0,78
Mink	0,76

Tabulka 8.5: KCNQ1.

Slovo	Podobnost
Phorodon	0,88
damson	0,82
aphididae	0,79
• calceolariae	0,79
• bryani	0,79
asclepiadis	0,79
viburni	0,78
• calamistis	0,78
bicincta	0,78
• volucre	0,78

Tabulka 8.6: humuli.

Slovo	Podobnost
distension	0,60
defecation	0,59
tenesmus	0,57
anismus	0,57
void	0,56
tension	0,56
defecatory	0,55
nonrelaxing	0,55
defaecation	0,54
• urination	0,54

Tabulka 8.7: straining.

Slovo	Podobnost
latipes	0,89
medaka	0,86
Medaka	0,79
minutillus	0,76
ricefish	0,73
melastigma	0,72
dancena	0,70
Adrianichthyidae	0,70
Qurt	0,67
Lhcgrbb	0,66

Tabulka 8.8: Oryzias.

Slovo	Podobnost
lactate	0,69
Dehydrogenase	0,64
C_line	0,59
LDH	0,59
Accusport	0,59
Malate	0,59
pyruvate	0,58
GLDH	0,56
Pyruvate	0,56
deshydrogenase	0,56

Tabulka 8.9: Lactate.

Slovo	Podobnost
Paravertebral	0,79
epidural	0,75
paraspinal	0,74
prevertebral	0,70
extradural	0,69
lumbosacral	0,68
intercostal	0,67
intraspinal	0,67
lumbar	0,66
thoracal	0,66

Tabulka 8.10: paravertebral.

Slovo	Podobnost
AP5	0,66
phosphonovaleric	0,63
phosphonopentanoic	0,59
phosphonovalerate	0,58
CNQX	0,57
DNQX	0,56
7CKA	0.56
AAM077	0,56
Ro25-6981	0,55
pneumovirus	0,55

Tabulka 8.11: APV.

Slovo	Podobnost
BTG1	0,77
TIS21	0,74
BTG2TIS21	0,74
CCNG2	0,72
GADD45alpha	0,72
iASPP	0,70
FOXM1	0,70
KLF8	0,69
• MiRP1	0,69
p33ING1	0,69

Tabulka 8.12: BTG2.

Slovo	Podobnost
protuberant	0,68
crease	0,66
protuberance	0,65
concavity	0,65
bulbous	0,65
• cuticularised	0,64
indent	0,64
protrude	0,63
hairline	0,63
• fontanel	0,63

Tabulka 8.13: dimple.

Slovo	Podobnost
appetite	0,85
satiation	0,82
hunger	0,78
satiate	0,73
anorectic	0,69
orexigenic	0,68
overeat	0,67
ghrelin	0,67
hyperphagia	0,667
anorexigenic	0,66

Tabulka 8.14: satiety.

Slovo	Podobnost
HDPE	0,88
LLDPE	0,77
layflat	0,70
rHDPE	0,68
EPDM	0,67
EVOH	0,66
PVDC	0,65
compatibilizer	0,65
terephthalate	0,65
terephtalate	0,65

Tabulka 8.15: LDPE.

Slovo	Podobnost
premaxilla	0,87
vomer	0,82
nasomaxillary	0,79
retrusion	0,78
midfacial	0,78
mid-face	0,77
mandibular	0,76
mandible	0,76
retrognathia	0,76
palatal	0,76

Tabulka 8.16: premaxillary.

Kapitola 9

Závěr

Metody hlubokého učení získaly v posledních letech velkou pozornost. Jedná se učící algoritmy, které automaticky odhalují určité vlastnosti dat bez potřeby dalších informací. Tento koncept bývá obvykle rychlejší než u algoritmů s učitelem nebo u informovaných metod při zpracovávání velkých nestrukturovaných korpusů. Obzvlášť v dnešní době, kdy množství biomedicínské literatury závratně roste a stává se stále náročnější ručně udržovat znalostní základny aktuální, tyto algoritmy mohou najít uplatnění.

Cílem práce bylo prostudovat metody měření sémantické podobnosti slov, analyzovat vhodné jazykové nástroje a na základě zjištěných poznatků vytvořit systém, který pro zadané slovo dokáže automaticky najít slova sémanticky příbuzná. Oblast dat byla specifikována na biomedicínské texty.

Při realizaci tohoto systému jsem využil nástroj Word2Vec, abych otestoval jeho schopnost identifikovat vztahy z nestrukturovaného textu. Word2Vec umožňuje různá nastavení, jejichž vliv na kvalitu výstupu jsem poté diskutoval. Testování výsledné reprezentace vektorového prostoru ukázalo, že dokáže určovat synonyma v dané doméně, jak požadovalo zadání. Úspěšnost tohoto úkolu činila 93 %. Při porovnávání modelu s dalšími prostředky dosahoval tento srovnatelných výsledků, přestože byl vytvořen z menšího objemu dat. Pokud bych měl k dispozici srovnatelně rozsáhlý korpus, nejspíše by došlo k překonání konkurence. To může být námět na pokračování práce.

Výsledky nemají dostatečnou kvalitu pro automatizaci znalostních bází a ontologií, mohou však pomoci k řešení řady problémů v biomedicínských studiích, které těží z předpočítaných reprezentací vektorů slov a z jazykových modelů. Celá práce může sloužit jako odrazový můstek pro další vyvíjené aplikace v této doméně. Navazující výzkum by se měl zaměřit na způsoby, jak kombinovat Word2Vec s prostředky biomedicínských ontologií a vytvořit tak hybridní systémy s větší přesností a flexibilitou, než zde uvedený přístup.

Ačkoliv je Word2Vec efektivní implementací pro svou schopnost identifikovat vztahy v textových datech bez dalších doménových znalostí, na základě kterých vytváří vektorové reprezentace, existují problémy, pro které se nehodí. Tento případ nastal při úkolu porovnávání vět konference SemEval, kde jsem také uvedl důvody průměrného umístění. Dosažení lepšího výsledku při porovnávání vět vyžaduje využití dalších nástrojů a metod.

Mezi přínosy této práce považuji také otestování 9 různých tokenizátorů. Vyhodnotil jsem jejich různá chování v několika scénářích a uvedl hlavní rozdíly mezi nimi. Ve výsledku jsem tedy čtenáři poskytl informace pro výběr vhodného nástroje k tokenizaci slov, který by měl vyhovovat jeho potřebám.

Literatura

[1] Agah, A.: *Medical Applications of Artificial Intelligence*. CRC Press, 2013.

[2] Alias-i: LingPipe.
URL http://alias-i.com/lingpipe/

[3] Apache Software Foundation: OpenNLP.
URL https://opennlp.apache.org/

[4] Bird, S.; Klein, E.; Loper, E.: *Natural Language Processing with Python*. O'Reilly Media, 2009.

[5] Brill, E.: A Simple Rule-based Part of Speech Tagger. In *Proceedings of the Workshop on Speech and Natural Language*, Association for Computational Linguistics, 1992, ISBN 1-55860-272-0, s. 112–116.

[6] Buscaino, B.: Trend Analysis in Biomedical Texts via Vector Space Model Synonym Extraction. Technická zpráva, Lawrence Livermore National Laboratory (LLNL), Livermore, CA, 2014.

[7] Gawron, J. M.: Semantic Similarity & Word distribution. Technická zpráva, San Diego State University, Department of Linguistics, 2011.

[8] Jones, K. S.; Walker, S.; Robertson, S. E.: A probabilistic model of information retrieval: development and comparative experiments: Part 1. *Information Processing & Management*, ročník 36, č. 6, 2000: s. 779–808.

[9] Lashkari, A. H.; Mahdavi, F.; Ghomi, V.: A boolean model in information retrieval for search engines. In *Information Management and Engineering, 2009. ICIME'09. International Conference on*, IEEE, 2009, s. 385–389.

[10] Lee, M. C.; Zhang, J. W.; Lee, W. X.; aj.: Sentence Similarity Computation Based on POS and Semantic Nets. In *INC, IMS and IDC, 2009. NCM'09. Fifth International Joint Conference on*, IEEE, Aug 2009, ISBN 978-0-7695-3769-6, s. 907–912.

[11] Li, Y.; Li, H.; Cai, Q.; aj.: A novel semantic similarity measure within sentences. In *Computer Science and Network Technology*, IEEE, Dec 2012, ISBN 978-1-4673-2963-7, s. 1176–1179.

[12] Lin, D.: An Information-Theoretic Definition of Similarity. In *Proceedings of the Fifteenth International Conference on Machine Learning*, Morgan Kaufmann Publishers Inc., 1998, ISBN 1-55860-556-8, s. 296–304.

[13] Liu, X.; Zhou, Y.; Zheng, R.: Sentence Similarity based on Dynamic Time Warping. In *Semantic Computing, 2007. ICSC 2007. International Conference on*, Sept 2007, ISBN 978-0-7695-2997-4, s. 250–256.

[14] Lund, K.; Burgess, C.: Producing high-dimensional semantic spaces from lexical co-occurrence. *Behavior Research Methods, Instruments, & Computers*, ročník 28, č. 2, 1996: s. 203–208, ISSN 0743-3808.

[15] Manning, C. D.; Raghavan, P.; Schütze, H.: *Introduction to information retrieval*, ročník 1. Cambridge university press Cambridge, 2008.

[16] McCallum, A. K.: MALLET: A Machine Learning for Language Toolkit, 2002. URL http://mallet.cs.umass.edu

[17] Mikolov, T.; Chen, K.; Corrado, G.; aj.: Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[18] Mikolov, T.; Sutskever, I.; Chen, K.; aj.: Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, 2013, s. 3111–3119.

[19] Ng, H. T.; Zelle, J.: Corpus-based approaches to semantic interpretation in NLP. *AI magazine*, ročník 18, č. 4, 1997: str. 45.

[20] Programering: Mahout similarity algorithm. 2014. URL http://www.programering.com/a/MTNxIzMwATk.html

[21] Řehůřek, R.; Sojka, P.: Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, ELRA, Květen 2010, s. 45–50.

[22] Research at Google: Natural Language Processing [online]. [cit. 2014-12-22]. URL http://research.google.com/pubs/NaturalLanguageProcessing.html

[23] Sochrová, M.: *Český jazyk v kostce pro střední školy*. Fragment, 1999.

[24] Toutanova, K.; Klein, D.; Manning, C. D.; aj.: Feature-rich Part-of-speech Tagging with a Cyclic Dependency Network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, Association for Computational Linguistics, 2003, s. 173–180.

[25] U.S. National Library of Medicine: MedPost/SKR Part of Speech Tagger. URL http://metamap.nlm.nih.gov/MedPostSKRTagger.shtml

[26] U.S. National Library of Medicine: MetaMap. URL http://metamap.nlm.nih.gov/

[27] Wallis, S.: Searching treebanks and other structured corpora. 2008.

[28] Wiemer-Hastings, P.; Wiemer-Hastings, K.; Graesser, A.: Latent semantic analysis. In *Proceedings of the 16th international joint conference on Artificial intelligence*, Citeseer, 2004, s. 1–14.

[29] Wikipedia: Natural language processing [online]. Poslední modifikace: 2014-11-10, [cit. 2014-12-22].
URL http://en.wikipedia.org/wiki/Natural_language_processing

[30] Yoshimasa Tsuruoka: Genia Tagger.
URL http://www.nactem.ac.uk/GENIA/tagger/

[31] Zhang, Y.; Deng, K.: Similarity Measure Based on Improved Optimal Assignment Model. In *Proceedings of the 2010 Second International Conference on Intelligent Human-Machine Systems and Cybernetics-Volume 01*, IEEE Computer Society, 2010, s. 125–128.

Příloha A

Výstupy jednotlivých tokenizátorů

V této části budou uvedeny výstupy každého z testovaných tokenizátorů pro testovací data. Symbol ▵značí hranice sousedních tokenů.

Vstupní data:

```
-5 3.14 11:20 17-19 3/4 51,000 50%
-67.8 11:20-11:30 3.0/4.0 1.5-2.5% 44.9+/-3.0
20th 1990s cm2
etc. p.a. PhDr. I.Q.
woman's Texas' 1990's isn't I'll o'clock O'Doole 'right way' 'Good morning'
disease-causing Hi-Tech ninety-nine
m/s male/female R/W 16/4/2015
(atom) (44) (DNA) (DNA and RNA) [RNA]
H20 H2SO4 3'-deoxyATP 5,6-dihydrodiol-3-methyl-2-oxo-1,2-dihydroquinoline
5'-AAAGTCT-3'
```

NLTK TreebankWordTokenizer

```
-5▵3.14▵11:20▵17-19▵3/4▵51,000▵50%
-67.8▵11:20-11:30▵3.0/4.0▵1.5-2.5▵%▵44.9+/-3.0
20th▵1990s▵cm2
etc▵.▵p.a▵.▵PhDr▵.▵I.Q▵.
woman▵'s▵Texas▵'▵1990▵'s▵is▵n't▵I▵'ll▵o'clock▵O'Doole▵'right▵way▵'▵'Good▵
   morning'
disease-causing▵Hi-Tech▵ninety-nine
m/s▵male/female▵R/W▵16/4/2015
(▵atom▵)▵(▵44▵)(▵DNA▵)(▵DNA▵and▵RNA▵)[▵RNA▵]
H20▵H2SO4▵3'-deoxyATP▵5,6-dihydrodiol-3-methyl-2-oxo-1,2-dihydroquinoline
5'-AAAGTCT-3▵'
```

NLTK WhitespaceTokenizer

```
-5▵3.14▵11:20▵17-19▵3/4▵51,000▵50%
-67.8▵11:20-11:30▵3.0/4.0▵1.5-2.5%▵44.9+/-3.0
20th▵1990s▵cm2
etc.▵p.a.▵PhDr.▵I.Q.
woman's▵Texas'▵1990's▵isn't▵I'll▵o'clock▵O'Doole▵'right▵way'▵'Good▵
   morning'
```

disease-causing␣Hi-Tech␣ninety-nine
m/s␣male/female␣R/W␣16/4/2015
(atom)␣(44)␣(DNA)(DNA␣and␣RNA)[RNA]
H20␣H2SO4␣3'-deoxyATP␣5,6-dihydrodiol-3-methyl-2-oxo-1,2-dihydroquinoline
5'-AAAGTCT-3'

Stanford POS Tagger

-5␣3.14␣11:20␣17-19␣3/4␣51,000␣50␣%
-67.8␣11:20␣-11:30␣3.0␣/␣4.0␣1.5-2␣.5␣%␣44.9␣+␣/␣-3.0
20th␣1990s␣cm2
etc.␣p.a.␣PhDr␣.␣I.Q.
woman␣'s␣Texas␣'␣1990␣'s␣is␣n't␣I␣'ll␣o'clock␣O'Doole␣'␣right␣way␣'␣'␣Good␣
morning␣'
disease-causing␣Hi-Tech␣ninety-nine
m/s␣male/female␣R/W␣16/4/2015
-LRB-␣atom␣-RRB-␣-LRB-␣44␣-RRB-␣-LRB-␣DNA␣-RRB-␣-LRB-␣DNA␣and␣RNA␣-RRB-␣
-LSB-␣RNA␣-RSB-
H20␣H2SO4␣3␣'␣-␣deoxyATP␣5,6-dihydrodiol-3-methyl-2-oxo-1␣,2␣-␣
dihydroquinoline
5␣'␣-␣AAAGTCT-3␣'

Mallet tokenizer

(žádný výstup)
(žádný výstup)
th␣s␣cm
etc␣p␣a␣PhDr␣I␣Q
woman␣s␣Texas␣s␣isn␣t␣I␣ll␣o␣clock␣O␣Doole␣right␣way␣Good␣morning
disease␣causing␣Hi␣Tech␣ninety␣nine
m␣s␣male␣female␣R␣W␣16␣4␣2015
atom␣TODO␣DNA␣DNA␣and␣RNA␣RNA
H␣0␣H␣SO␣deoxyATP␣dihydrodiol␣methyl␣oxo␣dihydroquinoline
AAAGTCT

OpenNLP tokenizer

-␣5␣3␣.␣14␣11␣:␣20␣17␣-␣19␣3␣/␣4␣51␣,␣000␣50␣%
-␣67␣.␣8␣11␣:␣20␣-␣11␣:␣30␣3␣.␣0␣/␣4␣.␣0␣1␣.␣5␣-␣2␣.␣5␣%␣44␣.␣9␣+␣/␣-␣3␣.␣0
20␣th␣1990␣s␣cm␣2
etc␣.␣p␣.␣a␣.␣PhDr␣.␣I␣.␣Q␣.
woman␣'␣s␣Texas␣'␣1990␣'␣s␣isn␣'␣t␣I␣'␣ll␣o␣'␣clock␣O␣'␣Doole␣'␣right␣way␣'␣'␣
Good␣morning␣'
disease␣-␣causing␣Hi␣-␣Tech␣ninety␣-␣nine
m␣/␣s␣male␣/␣female␣R␣/␣W␣16␣/␣4␣/␣2015
(␣atom␣)␣(␣44␣)␣(␣DNA␣)␣(␣DNA␣and␣RNA␣)␣[␣RNA␣]
H␣20␣H␣2␣SO␣4␣3␣'␣-␣deoxyATP␣5␣,␣6␣-␣dihydrodiol␣-␣3␣-␣methyl␣-␣2␣-␣oxo␣-␣1␣,␣
2␣-␣dihydroquinoline
5␣'␣-␣AAAGTCT␣-␣3␣'

Lingpipe

```
-5 3 . 14 11 : 20 17-19 3 / 4 51 , 000 50 %
-67 . 8 11 : 20-11 : 30 3 . 0 / 4 . 0 1 . 5-2 . 5 % 44 . 9 + / -3 . 0
20th 1990s cm2
etc . p . a . PhDr . I . Q .
woman's Texas' 1990's isn't I'll o'clock 0'Doole 'right way' 'Good morning'
disease-causing Hi-Tech ninety-nine
m / s male / female R / W 16 / 4 / 2015
( atom ) ( 44 ) ( DNA ) ( DNA and RNA ) [ RNA ]
H20 H2SO4 3'-deoxyATP 5 , 6-dihydrodiol-3-methyl-2-oxo-1 ,
   2-dihydroquinoline
5'-AAAGTCT-3'
```

Genia tagger

```
-5 3.14 11 : 20 17-19 3/4 51 , 000 50 %
-67.8 11 : 20-11 : 30 3.0/4.0 1.5-2.5 % 44.9+/-3.0
20th 1990s cm2
etc . p.a . PhDr . I.Q .
woman 's Texas ' 1990 's is n't I 'll o'clock 0'Doole 'right way ' 'Good
   morning'
disease-causing Hi-Tech ninety-nine
m/s male/female R/W 16/4/2015
( atom ) ( 44 ) ( DNA ) ( DNA and RNA ) [ RNA ]
H20 H2SO4 3'-deoxyATP 5 , 6-dihydrodiol-3-methyl-2-oxo-1 ,
   2-dihydroquinoline
5'-AAAGTCT-3 '
```

MetaMap

```
- 5 3 . 14 11 : 20 17 - 19 3 / 4 51 , 000 50 %
- 67 . 8 11 : 20 - 11 : 30 3 . 0 / 4 . 0 1 . 5 - 2 . 5 % 44 . 9 + / - 3 . 0
20th 1990s cm2
etc . p . a . PhDr . I . Q .
woman's Texas ' 1990's isn ' t I ' ll o ' clock 0 ' Doole ' right way ' ' '
   Good morning '
disease - causing Hi - Tech ninety - nine
m / s male / female R / W 16 / 4 / 2015
( atom ) ( 44 ) ( DNA ) ( DNA and RNA ) [ RNA ]
H20 H2SO4 3 ' - deoxyATP 5 , 6 - dihydrodiol - 3 - methyl - 2 - oxo - 1 ,
   2 - dihydroquinoline
5 ' - AAAGTCT - 3 '
```

MedPost/SKR POS Tagger

```
- 5 3 . 14 11 : 20 17 - 19 3 / 4 51 , 000 50 %
- 67 . 8 11 : 20 - 11 : 30 3 . 0 / 4 . 0 1 . 5 - 2 . 5 % 44 . 9 + / - 3 . 0
20th 1990s cm2
etc . p . a . PhDr . I . Q .
woman ' s Texas ' 1990 ' s isn ' t I ' ll o ' clock 0 ' right way ' ' '
   Good morning '
```

desease␣-␣causing␣Hi␣-␣Tech␣ninety␣-␣nine
m␣/␣s␣male␣/␣female␣R␣/␣W␣16␣/␣4␣/␣2015
(␣atom␣)␣(␣44␣)(␣DNA␣)␣(␣DNA␣and␣RNA)␣[␣RNA␣]
H2O␣H2SO4␣3␣'␣-␣deoxyATP␣5␣,␣6␣-␣dihydrodiol␣-␣3␣-␣methyl␣-␣2␣-␣oxo␣-␣1␣,␣2␣
-␣dihydroquinoline
5␣'␣-␣AAAGTCT␣-␣3␣'

Příloha B

Obsah přiloženého DVD

Přiložené DVD obsahuje následující soubory:

- zdrojové kódy souborů pro vytváření modelů a manipulaci s nimi
- vytvořený model
- skript k úkolu SemEval
- soubor README
- elektronickou verzi této zprávy
- plakát prezentující práci